UNIVERSITATEA "POLITEHNICA" DIN TIMIŞOARA
FACULTATEA DE HIDROTEHNICĂ

TEZĂ DE DOCTORAT

MODELAREA NUMERICĂ A SISTEMELOR COMPLEXE DE EXPLOATARE A APELOR SUBTERANE.

METODA ELEMENTELOR DE FRONTIERĂ PENTRU MODELAREA NUMERICĂ A SISTEMELOR DE EXPLOATARE A APELOR SUBTERANE

Conducător ştiinţific,
Prof.dr.ing. Ioan David

Autor,
Şef lucr.ing. Gabriel Eleş

- 2000 -

CAPITOLUL I.

INTRODUCERE

1.1 Considerații generale asupra metodelor de modelare, obiective.

În general în condițiile naturale întâlnite în practică mișcările prin medii poroase sunt tridimensionale. Caracterul tridimensional al mișcării se manifestă și în cazul în care mediul prezintă aceeași configurație geometrică în planuri paralele cu un plan dat, cum este cazul mediilor poroase stratificate. Ținând cont de complexitatea problemelor tridimensionale se caută însă simplificarea mișcărilor prin care problemele tridimensionale se pot reduce la probleme bidimensionale echivalente.

În cazul problemelor bidimensionale (mișcările plane) componentele vitezei și presiunea vor depinde numai de coordonatele x, y față de un sistem de axe luat într-un plan, paralel cu planul mișcării.

Aceste mărimi fizice de bază vor satisface un anumit sistem de relații matematice (diferențiale, integrale etc.) care constituie modelul matematic general al mișcării prin medii poroase.

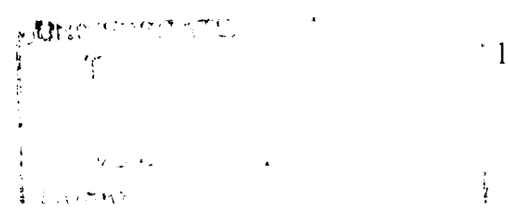
Problemele se reduc la determinarea unei funcții necunoscute, de regulă înălțimea piezometrică, care satisface anumite condiții la limită pe frontiera domeniului în care se studiază mișcarea.

Modelarea matematică a mișcării apei prin medii poroase s-a dezvoltat în trei direcții de formulare [25]: diferențială, integrală și variațională (fig.1).

Prima formulare și anume cea diferențială necesită rezolvarea unei ecuații cu derivate parțiale de ordinul doi cu două variabile independente, domeniul de definire și condițiile la limită pe frontiera domeniului fiind date. Aici se încadrează ecuațiile de tip Laplace, Poisson, respectiv ecuațiile de tip eliptic general care reprezintă modelul matematic al mișcării în medii poroase omogene, respectiv neomogene.

A doua formulare este așa numita formularea variațională care se bazează pe găsirea unei funcționale ce se minimizează. Potrivit principiului variațional, rezolvarea unei ecuații diferențiale într-un anumit domeniu și cu anumite condiții la limită este echivalentă cu minimizarea în acel domeniu a unei mărimi funcționale corespunzătoare ecuației diferențiale și condițiilor la limită date.

A treia formulare, formularea integrală, necesită rezolvarea unei ecuații integrale. Aceasta a fost considerată de către Hilbert pentru a ilustra cercetările sale în domeniul ecuațiilor integrale. Ulterior problema a fost tratată pe diferite căi



utilizându-se de exemplu ecuații integrale de tip Fredholm la care se ajunge fie direct fie prin transformări integrale Green şi care stau la baza metodei numerice cu elemente de frontieră.

Soluționarea unei probleme de mişcare, indiferent de formulare, presupune în ultimă instanță determinarea funcției necunoscute, în cazul de față înălțimea piezometrică (potențial hidraulic), în domeniul mişcării. Distribuția acestei funcții în domeniu este univoc determinată de condițiile restrictive impuse de ecuațiile fundamentale (diferențiale, integrale etc.) specifice tipului de formulare. Obținerea unei soluții pe cale analitică este posibilă numai în cazuri particulare de mişcări respectiv domenii şi condiții la limită simple. În general este necesar deci să se recurgă la metode numerice care permit obținerea unei soluții acceptabile în cele mai complexe condiții. Principalele metode numerice care s-au dezvoltat pe parcursul anilor, în mod deosebit în ultimele două decenii sunt:

- Metoda diferențelor finite (MEDIF)
- Metoda volumelor finite (MEVFIN)
- Metoda elementelor finite (MEFIN)
- Metoda elementelor de frontieră (MEFRO)

Cuplarea acestor metode cu tipurile de formulare a problemei la limită se poate urmări în fig.(1).

În ceea ce priveşte rezolvarea numerică efectivă, aceasta presupune întotdeauna un anumit tip de discretizare: discretizarea întregului domeniu în primele trei cazuri (MEDIF, MEVFIN şi MEFIN), respectiv discretizarea numai a frontierei domeniului în cazul MEFRO.

Pentru fiecare dintre aceste metode există la ora actuală programe performante de largă circulație pe plan mondial cum ar fi: MODFLOW [59] şi ASM [41], [42] pentru metoda volumelor finite şi AQUA [57] pentru metoda elementelor finite precum şi multe altele care pot fi urmărite în Catalogul periodic al Scientific Software group-Washington, Environmental Software Publications)

În ceea ce priveşte MEFRO lipsesc programe complexe pentru modelarea mişcării apei subterane de anvergura celor de mai sus.

Obiectivul principal al lucrării îl constituie completarea posibilităților de utilizare a MEFRO pentru modelarea mişcării apelor subterane prin noi programe care valorifică avantajele acestei metode mai ales în anumite condiții speciale de mişcare. Pentru a evidenția aceste posibilități cu elemente de noutate este necesar să se reamintească faptul că metodele MEDIF, MEVFIN şi MEFIN pe care le vom numi în continuare metode standard, presupun întotdeauna discretizarea domeniului mişcării ceea ce conduce la erori care pot ajunge la valori foarte mari mai ales în zone locale unde relația matematică dintre înălțimea piezometrică şi flux are un puternic caracter neliniar, fiind de tip logaritmic sau polar (din punct de vedere matematic singular). Asemenea zone se întâlnesc mai ales în vecinătatea diferitelor tipuri de captare subterană (puțuri, drenuri etc.) [27], [28], [29], [30], [31].

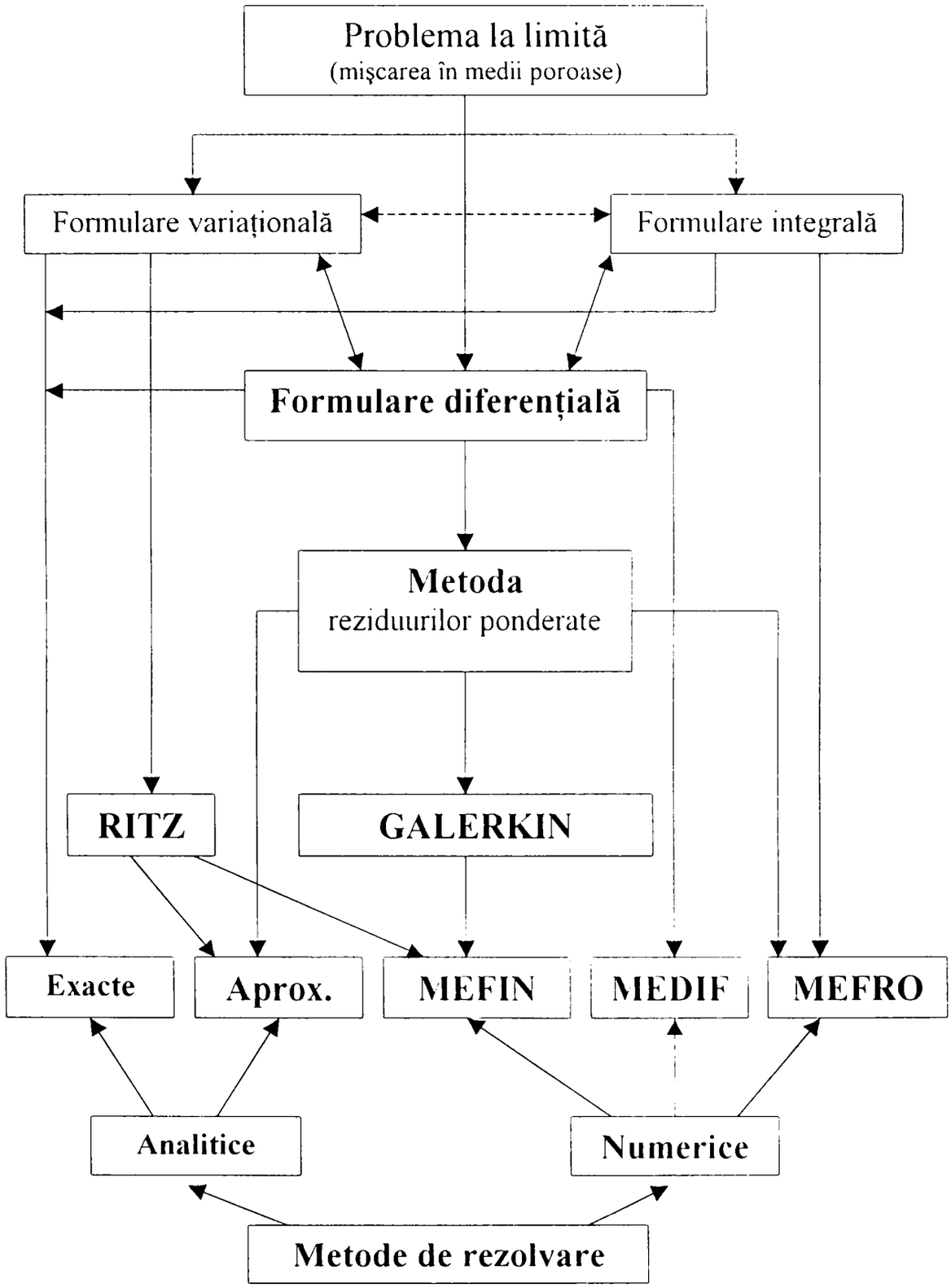


Fig.1 Formulări şi metode matematice de rezolvare a mişcării prin medii poroase (după David [25])

În cadrul lucrării în capitolul IV se prezintă o analiză detaliată a erorilor MEDIF, MEFIN cu anumite situații reprezentative de singularități pentru care există posibilitatea comparării rezultatelor cu soluții analitice exacte. În cadrul acestei analize se prezintă şi unele rezultate cu MEFRO (paragraful 4.3.1) pentru a avea o imagine completă asupra erorilor comparative a celor patru metode, metodele standard (MEDIF, MEVFIN, MEFIN) şi MEFRO.

Această analiză comparativă a diferitelor metode numerice scoate în evidență oportunitatea utilizării metodei elementelor de frontieră (MEFRO) mai ales în zonele unde mişcarea are un caracter singular. Pe această concluzie preliminară, bazată pe o serie de rezultate publicate, inclusiv cu contribuții ale autorului şi a conducătorului tezei s-au fixat obiectivele principale ale lucrării.

- Elaborarea unui program complex bazat pe metoda elementelor de frontieră care să conțină principalele obiecte ale unui sistem de captare subterană ce se întâlnesc în practică;
- Verificarea sistematică a programului elaborat prin exemple de tip singularități reprezentative (puțuri, puțuri imperfecte, drenuri, subdomenii);
- Sistematizarea programului într-un software unitar cu posibilități de utilizare relativ uşoară în activitatea de proiectare.

Toate aceste aspecte vor fi prezentate pe larg în capitolele următoare ale lucrării.

CAPITOLUL II

PREZENTAREA SINTETICĂ A ECUAȚIILOR FUNDAMENTALE ALE MIȘCĂRII PRIN ACVIFERE ȘI A FORMULĂRII MATEMATICE A PROBLEMELOR LA LIMITĂ

În cadrul acestui capitol se va prezenta o scurtă sinteză a principalelor metode de reprezentare şi formulare a problemelor de mişcare a apei în acvifere.

2.1. Ecuațiile mişcării apei prin medii poroase.

Forma diferențială (locală) a legii lui Darcy pentru mişcări unidimensionale este:

$$V = -k\frac{dh}{dl}$$

Pentru cazul mai general al unei mişcări bidimensionale şi tridimensionale legea lui Darcy generalizată are forma:

$$\vec{v} = -k\nabla h$$

sau după componente:

$$v_x = -k\frac{\partial h}{\partial x}; \qquad v_y = -k\frac{\partial h}{\partial y}; \qquad v_z = -k\frac{\partial h}{\partial z}; \qquad h = \frac{p}{\rho g}$$

în cazul tridimensional, respectiv cu $V_z = 0$ în cazul bidimensional

semnificația mărimilor care intervin în relațiile de mai sus este următoarea:

$h = h(M)$ - funcția de sarcină hidraulică (înălțime piezometrică sau potențial hidraulic);

$k = k(M)$ - coeficientul de filtrație;

v - viteza de filtrație;

p - presiunea în punctul M;

z - poziția lui M în planul vertical;

M - un punct arbitrar în domeniul mişcării, având coordonatele x,y,z;

∇ - simbolul nabla.

Mediul poros în care coeficientul de filtrație variază de la un punct la altul se numeşte **mediu poros neomogen**.

În cazul în care coeficientul de filtrație prezintă variații direcționale, respectiv pentru acelaşi gradient hidraulic viteza diferă pentru două direcții diferite, mediul poros poartă denumirea de **mediu anizotrop**. În acest caz coeficientul de filtrație este o mărime tensorială, iar legea lui Darcy devine:

$$\vec{v} = -\overrightarrow{\overrightarrow{k}} \cdot \nabla h$$

unde $\overrightarrow{\overrightarrow{k}}$ este tensorul coeficientului de filtrație, un tensor simetric de ordinul doi.

În cazul bidimensional (mişcare în planul orizontal) spre exemplu dacă axele Ox şi Oy coincid cu axele principale avem relațiile:

$$V_x = k_{xx}\frac{\partial h}{\partial x};$$

$$V_y = k_{yy}\frac{\partial h}{\partial y};$$

Ecuațiile de mişcare se completează cu ecuația de continuitate:

$$\nabla\left(m\vec{V}\right) = \varepsilon \quad \text{respectiv} \quad \nabla\left(h\vec{V}\right) = \varepsilon$$

$$\frac{d}{dx}(mV_x) = \varepsilon \quad ; \quad \frac{d}{dx}(hV_x) = \varepsilon$$

pentru cazul unidimensional al mişcării sub presiune respectiv cu nivel liber şi:

$$\frac{\partial}{\partial x}(mV_x) + \frac{\partial}{\partial y}(mV_y) = \varepsilon \quad ; \quad \frac{\partial}{\partial x}(hV_x) + \frac{\partial}{\partial y}(hV_y) = \varepsilon$$

în cazul bidimensional.

ε - reprezintă aportul de debit distribuit care alimentează stratul acvifer.

2.2 Formularea diferențială a problemelor de filtrație.

Formularea diferențială a unei probleme de filtrație are la bază ecuațiile fundamentale prezentate mai sus la care se adaugă :

- definirea domeniului;
- condițiile la limită.

Având în vedere ecuația de continuitate şi ecuația lui Darcy generalizată

$$\begin{cases} \vec{v} = -k \cdot \nabla h \\ \nabla \cdot \vec{v} = 0 \end{cases}$$

prin înlocuire se obține ecuația generală a mişcării;

$$\nabla \cdot (k \nabla h) = 0$$

în formă vectorială, respectiv:

$$\frac{\partial}{\partial x}\left(k\frac{\partial h}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial h}{\partial y}\right) + \frac{\partial}{\partial z}\left(k\frac{\partial h}{\partial z}\right) = 0$$

în formă scalară, mişcare tridimensională, ceea ce reprezintă o ecuație diferențială cu derivate parțiale de tip eliptic.

În aplicațiile practice, de mişcări în acvifere extinse în plan orizontal sunt reprezentative mişcările bidimensionale. Aceste mişcări pot fi: mişcări în strat acvifer sub presiune şi mişcări în strat acvifer cu nivel liber.

a) Mişcări bidimensionale în strat acvifer sub presiune.

$$\vec{v} = -k\nabla h$$

$$\nabla(T\nabla h) = 0 \quad ; \quad T = km \ ;$$

$$\frac{\partial}{\partial x}\left(T\frac{\partial h}{\partial x}\right) + \frac{\partial}{\partial y}\left(T\frac{\partial h}{\partial y}\right) = 0 \quad , \quad h = h(x,y) \ , \quad T = T(x,y)$$

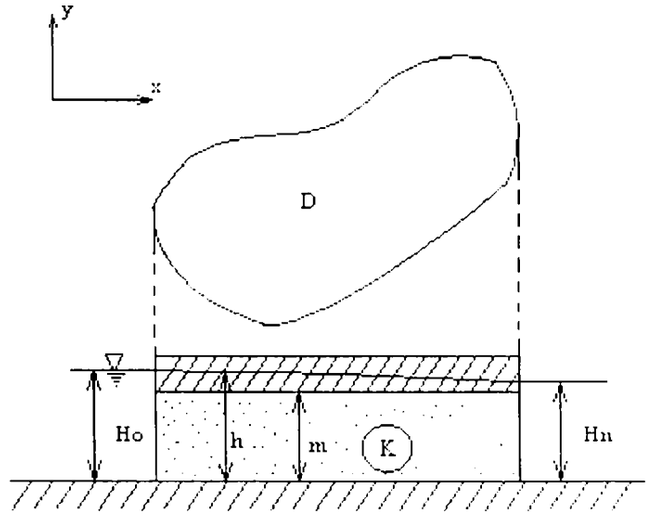


Fig 2.1 Strat acvifer cu nivel limitat

h - înălțimea piezometrică;

T - transmisivitatea domeniului

b) Mişcări bidimensionale cu nivel liber.

În baza ipotezelor Dupuit - Forchheimer şi ţinând cont de ecuaţiile de mai sus avem:

$$v_z \approx 0 \quad ; \quad u = -k\nabla h \; ; \quad \nabla\left(h\vec{v}\right) = 0$$

respectiv ecuaţia generală:

$$\frac{\partial}{\partial x}\left(kh\frac{\partial h}{\partial x}\right) + \frac{\partial}{\partial y}\left(kh\frac{\partial h}{\partial y}\right) = 0$$

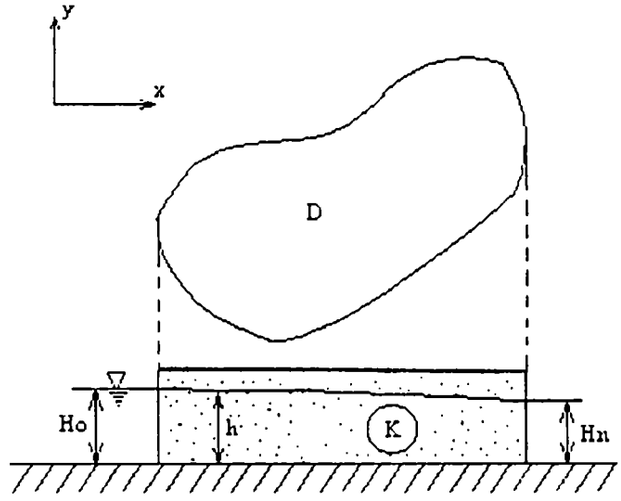


Fig 2.2 Strat acvifer cu nivel liber

Dacă mediul poros este omogen (k=const) atunci se poate introduce potenţialul vitezelor.

$-\varphi = -T \cdot h + C$; pentru strat acvifer sub presiune;

$-\varphi = -T\frac{h^2}{2} + C$; pentru strat acvifer cu nivel liber.

Cu ajutorul acestor funcţii de potenţial rezultă:

$\vec{v} = -T\nabla h = \nabla\varphi$; $T = km$, transmisivitatea domeniului

$\nabla \cdot \vec{v} = \nabla \cdot \left(\nabla \cdot \varphi\right) = \Delta\varphi$

$\nabla \cdot \vec{v} = 0 \quad \Rightarrow \quad \Delta\varphi = 0$ sau $\frac{\partial^2\varphi}{\partial x^2} + \frac{\partial^2\varphi}{\partial y^2} + \frac{\partial^2\varphi}{\partial z^2} = 0$

Unicitatea soluției, adică funcția $h(x,y,z)$ respectiv $\varphi(x,y,z)$ corespunzătoare unei anumite probleme de filtrație presupune cunoaşterea domeniului precum şi a condițiilor la limită pentru funcțiile h sau φ căutate.

2.2.1 Condiții la limită formularea diferențială a problemelor de filtrație.

Cele mai uzuale condiții la limită sunt:

a) Condiții de tip Dirichlet, caz în care este dată funcția pe frontiera C a domeniului

$h = h_s|_C \quad (h: D \to R)$

$\varphi = \varphi_s|_C \quad (\varphi: D \to R)$ în cazul mediilor omogene

C = frontiera domeniului

b) Condiții de tip Neumann, caz în care este dată derivata normală a funcției de-a lungul frontierei C

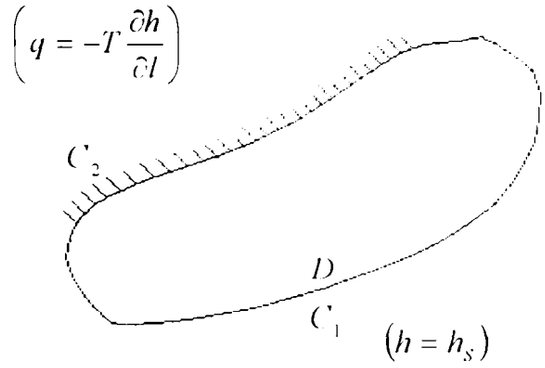
$-T\dfrac{\partial h}{\partial n} = q|_C \quad (h: D \to R) \quad ; \quad T = km$

$\dfrac{\partial \varphi}{\partial n} = q|_C \quad (\varphi: D \to R)$ în cazul mediilor omogene, C = frontiera domeniului

$q|_C = 0$ în cazul frontierei impermeabile

c) Condiții de tip mixt

În acest caz frontiera domeniului cuprinde porțiuni pe care este cunoscută fie valoarea funcției h, fie valoarea funcției q.



$C_1 \cup C_2$ = Frontiera domeniului

D = Domeniul miscarii

Fig. 2.3 Condiții de tip mixt

$h = h_S|_{C1}$; $-T\dfrac{\partial h}{\partial n} = q|_{C2}$ $T = km$, transmisivitatea domeniului

$(h: D \to R)$ şi $C_1 \cup C_2 = C$ =frontiera domeniului

respectiv:

$\varphi = \varphi_s|_{C1}$

$\dfrac{\partial \varphi}{\partial n} = q|_{C2}$

$\varphi: D \to R$ şi $C_1 \cup C_2 = C$ =frontiera domeniului

2.3 Formularea variațională

Se va considera mişcarea plană bidimensională într-un strat acvifer cu nivel sub presiune problema la limită de tip mixt.

În formulare diferențială avem:

$$\frac{\partial}{\partial x}\left(T\frac{\partial h}{\partial x}\right) + \frac{\partial}{\partial y}\left(T\frac{\partial h}{\partial y}\right) = -\varepsilon$$

$$h|_{C1} = h_0$$

$$-T\frac{\partial h}{\partial n}\bigg|_{C2} = q \quad ; \quad T = km$$

unde cu T s-a notat transmisivitatea mediului, ε reprezintă aport din exterior iar $C_1 \cup C_2 = C$ =frontiera domeniului

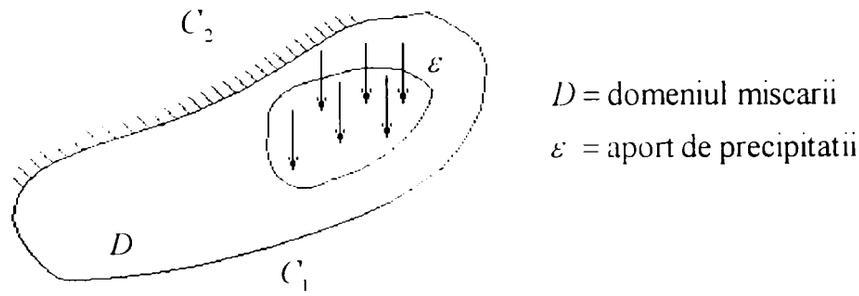


Fig. 2.4 Formularea variațională

Pentru formularea variațională echivalentă cu cea diferențială prezentată mai sus se consideră funcționala U definită prin:

$$U = \frac{1}{2}\iint_D \left[T\left(\frac{\partial h}{\partial x}\right)^2 + T\left(\frac{\partial h}{\partial y}\right)^2 \right] dxdy - \iint_D \varepsilon h dxdy + \int_{C1} qhdl =$$

$$\frac{1}{2}\iint_D T\nabla h \cdot \nabla h ds - \iint_D \varepsilon h dxdy + \int_{C1} qhdl \qquad ; \qquad T = km$$

Variațiile admisibile ale funcției h sunt determinate de condițiile pe frontiera domeniului adică, $\delta h = 0$ pe porțiunea de frontieră C_1 unde h este cunoscut (dat).

Variația funcționalei se poate exprima sub forma:

$$\delta U = \iint_D T\nabla h \cdot \nabla \varepsilon h dl - \iint_D \varepsilon \delta h dxdy + \int_{C1} q\delta_{hdl} \qquad ; \qquad T = km$$

Rezolvarea problemei la limită în formulare diferențială este echivalentă cu rezolvarea unei probleme variaționale care constă în determinarea funcției admisibile $h(x,y)$ care să minimizeze funcționala U, adică $\delta U = 0$ în domeniu.

Soluția h în formularea variațională se caută sub forma unei aproximări (metoda Ritz). Aplicarea acestei formulări va fi prezentată ceva mai detaliat în cazul prezentării metodei elementelor finite în cadrul capitolului III.

2.4 Formularea cu ajutorul ecuațiilor integrale.

Această formulare care stă la baza metodei elementelor de frontieră se poate aplica în special în cazul mediilor omogene. Aplicarea ei este posibilă însă şi în cazul mediilor omogene pe porțiuni. În principiu se porneşte şi în acest caz de la formularea diferențială a problemelor de filtrație. Luând exemplul mişcării bidimensionale, se are în vedere ecuația de tip Laplace sau Poisson:

$$\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2} = 0 \qquad ; \text{ ecuația de tip Laplace}$$

$$\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2} = -\frac{\varepsilon}{T} \qquad ; \text{ ecuația de tip Poisson} \qquad ; \qquad T = km$$

unde condițiile de pe frontiera domeniului sunt următoarele:

$$h\big|_{C_H} = h_0 \quad ; \quad \left.\frac{\partial h}{\partial n}\right|_{C_\Sigma} = 0 \quad ; \quad C = C_H \cup C_\Sigma = \text{frontiera domeniului}$$

Cu C_H, respectiv C_Σ s-au notat porțiunile de frontieră cu h dat, respectiv q_d dat care în cazul de față se consideră $q_\Sigma = 0$, adică C_Σ porțiune impermeabilă.

În continuare se folosește soluția fundamentală a ecuației de tip Laplace care este de forma:

$$h(z_1) = -\frac{1}{2\pi} \ln r(z_1, z_2)$$

soluție în care $h(z_1)$ reprezintă potențialul generat în punctul $z_1 \in D$ (D = domeniul mișcării) de către o sursă având intensitatea unitară amplasată în punctul $z_2 \in D$. Se observă că pentru simplificare un punct $M(x, y)$ al domeniului D s-a notat cu "z", $z \in D$

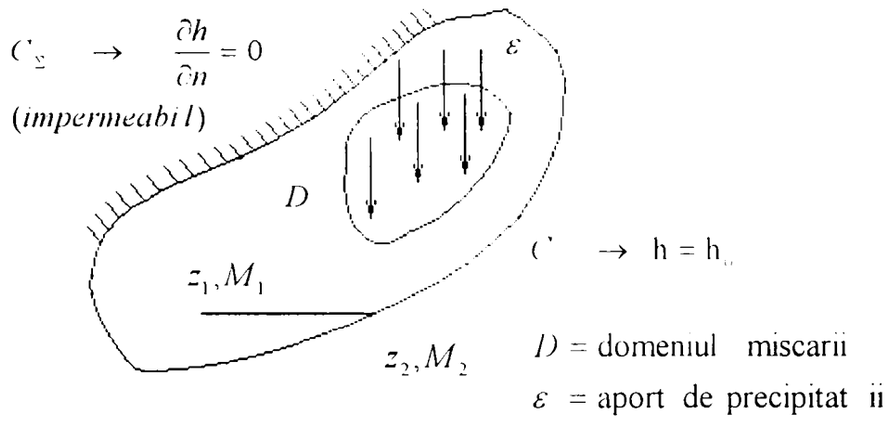


Fig. 2.5 Schema valabilă pentru formularea integrală

Cu ajutorul soluției fundamentale se poate exprima soluția generală a problemei sub formă integrală care în reprezentarea integrală indirectă este [3], [23], [25]:

$$h(z) = -\frac{1}{2\pi T} \int_C \psi(\zeta) \ln r(z, \zeta) dl - \frac{1}{2\pi T} \int_D \varepsilon(z^*) \ln r(z, z^*) d\Omega + c$$

unde:

$z, z^* \in D$ și $\zeta \in C$, $h(z)$ reprezintă înălțimea piezometrică în punctul $z \in D$ exprimată cu ajutorul unor surse fictive de distribuție pe conturul C având densitatea $\psi(\zeta)$, c fiind o constantă nedeterminată.

Prin derivarea expresiei înălțimii piezometrice $h(z)$ se poate exprima viteza de filtrație într-un punct z din domeniul mișcării după direcția normalei "n".

$$v(z) = -k \frac{\partial h}{\partial n} = \frac{k}{2\pi T} \int_C \psi(\zeta) \frac{\partial \ln r(z, \zeta)}{\partial n_z} dl + \frac{k}{2\pi T} \int_D \varepsilon(z^*) \frac{\partial \ln r(z, z^*)}{\partial n_z}$$

În cazul formulării directe expresia înălțimii piezometrice într-un punct din interiorul domeniului mişcării este de forma:

$$h(z) = -\frac{1}{2\pi T}\int_{D} \varepsilon(z^*)\ln r(z,z^*)d\Omega + \frac{1}{2\pi T}\int_{\Gamma}\left(\frac{\partial h}{\partial n}\ln r(z,\zeta) - h\frac{\partial \ln(z,\zeta)}{\partial n}\right)dl$$

Utilizând condițiile la limită, h-dat de Γ_1 şi $q = 0$ pe Γ_2 se ajunge în ambele cazuri la ecuații integrale din care se determină necunoscutele problemei[30] şi[31].

În cadrul formulării directe din ecuațiile integrale se determină valoarea lui h pe conturul Γ_2 şi $\frac{\partial h}{\partial n}$ pe conturul Γ_1. Astfel în final se cunosc valorile înălțimii piezometrice şi ale fluxului pe întreg conturul domeniului cu ajutorul cărora se poate calcula din reprezentarea de mai sus $h(z)$ într-un punct oarecare (z) al domeniului D.

În cazul formulării indirecte, pentru determinarea înălțimii piezometrice respectiv a vitezei de filtrație într-un punct din interiorul domeniului este necesară cunoaşterea distribuției surselor fictive ψ (densități ale surselor) de pe frontiera domeniului care se determină cu ajutorul condițiilor la limită, prin trecerea la limita reprezentărilor integrale pentru h respectiv v_n rezultând un sistem de ecuații integrale care se rezolvă de regulă numeric. Trecerea la limită împreună cu sistemul de ecuații rezultat vor fi prezentate în cadrul metodei elementelor de frontieră.

CAPITOLUL III

PREZENTAREA SINTETICĂ A METODELOR NUMERICE FOLOSITE ÎN HIDRAULICA SUBTERANĂ

În cazul în care rezolvarea problemelor de hidraulică subterană prin metode analitice devine dificilă din cauza complexității problemelor cum ar fi domenii cu geometrie complicată, medii neomogene sau dificultăți care intervin în cazul integrării ecuațiilor diferențiale ale problemei se recurge la metode numerice de rezolvare a problemelor.

Dintre metodele numerice cele mai des folosite şi răspândite se pot enumera următoarele [23], [25]: metoda diferențelor finite şi volumelor finite, metoda elementelor finite şi metoda elementelor de frontieră, fiecare din ele având specificul său.

3.1 Metoda diferențelor finite.

În cazul unui acvifer omogen rezolvarea unei probleme de filtrație constă în determinarea valorilor funcției h, respectiv φ, în nodurile sau ochiurile rețelei astfel încât aceste valori să satisfacă în orice nod interior relații echivalente cu ecuația generală a problemei respective, iar pe frontieră să fie respectate condițiile la limită specifice problemei considerate.

Bazele metodei[23], [25] constau în discretizarea domeniului (de regulă dreptunghiular) şi exprimarea derivatelor în funcție de valorile nodale. Metoda se bazează pe formularea diferențială:

- ecuația cu derivate parțiale

$$\frac{\partial^2 h}{\partial x^2}+\frac{\partial^2 h}{\partial y^2}=-\frac{\varepsilon}{T} \quad ; \quad T=km$$

- condițiile la limită fiind următoarele:

$$h|_{C_1}=g \quad \text{şi} \quad \left.\frac{\partial h}{\partial n}\right|_{C_2}=-f$$

unde: $h : D \to R$, este funcția potențial căutată, $C_1 \cup C_2$ = frontiera domeniului, g şi f sunt funcții date pe porțiunile de frontieră respective.

Rezolvarea problemei constă aşa cum s-a menționat mai sus în discretizarea domeniului de regulă într-o rețea de elemente rectangulare şi echidistante.

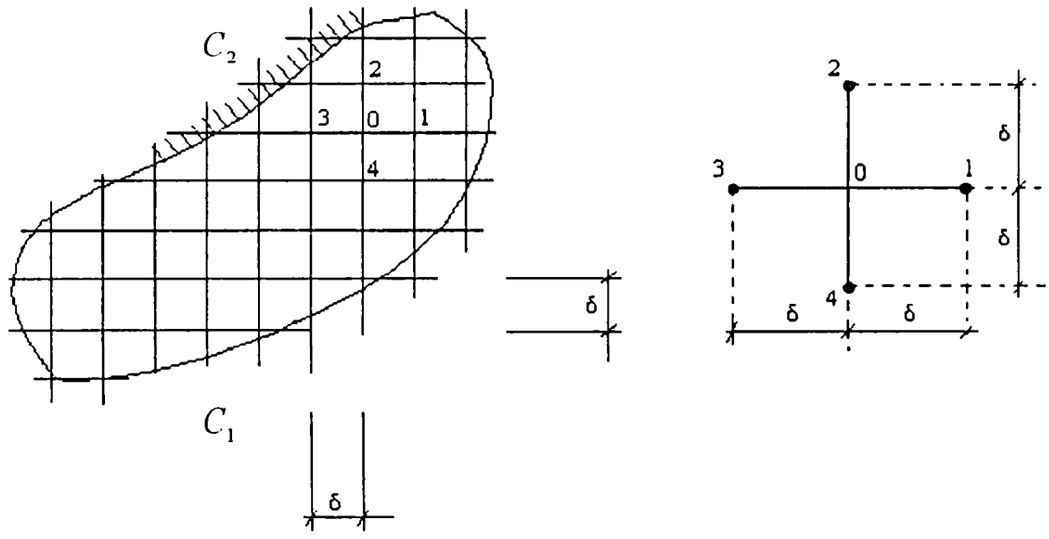


Fig. 3.1 Discretizarea domeniului(MEDIF)

Valorile funcției h în noduri punctele 1,...4 se pot exprima cu ajutorul valorilor şi a derivatelor calculate în punctul 0 prin dezvoltări în serie Taylor după direcțiile Ox şi Oy

$$h_1 = h_0 + \frac{\partial h}{\partial x}\delta + \frac{\partial^2 h}{\partial x^2}\frac{\delta^2}{2!} + \ldots$$

$$h_2 = h_0 + \frac{\partial h}{\partial y}\delta + \frac{\partial^2 h}{\partial y^2}\frac{\delta^2}{2!} + \ldots$$

$$h_3 = h_0 - \frac{\partial h}{\partial x}\delta + \frac{\partial^2 h}{\partial x^2}\frac{\delta^2}{2!} - \ldots$$

$$h_4 = h_0 - \frac{\partial h}{\partial y}\delta + \frac{\partial^2 h}{\partial y^2}\frac{\delta^2}{2!} - \ldots$$

Prin sumare şi ținându-se cont de ecuația generală, când $\varepsilon = 0$ se obține o relație echivalentă între valorile nodale ale înălțimii piezometrice h

$$h_0 = \frac{1}{4}(h_1 + h_2 + h_3 + h_4) + O(\delta^2) \cong \frac{1}{4}(h_1 + h_2 + h_3 + h_4)$$

relație care se scrie pentru toate nodurile interioare ale domeniului.

Ultima egalitate aproximativă "$\cong$" este o aproximație care s-a obținut din prima prin neglijarea termenilor proporționali cu "δ^2" care deci tind spre zero odată cu acesta.

În cazul acviferelor neomogene pentru obținerea relațiilor dintre valorile funcțiilor h sau φ se consideră cazul general al unui nod central, când în ochiurile învecinate coeficienții de filtrație au valori diferite $k^{(1)}, k^{(2)}, k^{(3)}, k^{(4)}$.

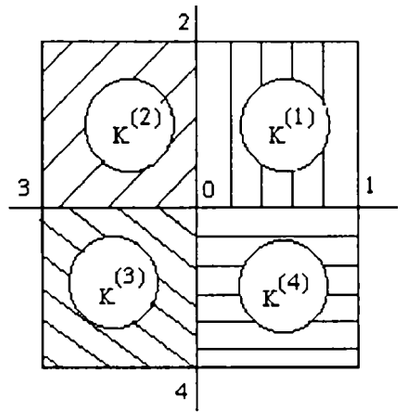


Fig 3.2 Schița nodurilor învecinate în cazul unui mediu neomogen

Conform semnificației fizice a funcției h, ea are valoare unică în fiecare nod al rețelei: h_0, h_1, h_2, h_3, h_4. Pe de altă parte funcția φ definită prin relația $\varphi = kh$ poate avea valori diferite în acelaşi nod, aceasta datorită discontinuității coeficientului de filtrație din ochiuri.

Astfel pentru punctul 1 se poate scrie:

$$\varphi_1^{(1)} = T_1^{(1)} h_1 \neq T_4^{(4)} h_1 = \varphi_1^{(4)}$$

$$\frac{\varphi_1^{(1)}}{T^{(1)}} = \frac{\varphi_1^{(4)}}{T^{(4)}} \; ; \; \frac{\varphi_2^{(1)}}{T^{(1)}} = \frac{\varphi_2^{(2)}}{T^{(2)}} \; ; \; \frac{\varphi_3^{(1)}}{T^{(2)}} = \frac{\varphi_3^{(3)}}{T^{(3)}} \; ; \; \frac{\varphi_4^{(3)}}{T^{(3)}} = \frac{\varphi_4^{(4)}}{T^{(4)}}$$

$$\frac{\partial \varphi^{(1)}}{\partial x} = \frac{\partial \varphi^{(2)}}{\partial x} \; ; \; \frac{\partial \varphi^{2)}}{\partial y} = \frac{\partial \varphi^{(3)}}{\partial y} \; ; \; \frac{\partial \varphi^{(3)}}{\partial x} = \frac{\partial \varphi^{(4)}}{\partial x} \; ; \; \frac{\partial \varphi^{(4)}}{\partial y} = \frac{\partial \varphi^{(1)}}{\partial y}$$

unde indicele superior indică cadranul.

Exprimând valorile funcției φ în nodurile 1-4 cu ajutorul valorilor şi a derivatelor calculate în nodul 0 prin dezvoltare în serie se obține:

după axa Ox:

$$\varphi_1^{(1)} = \varphi_0^{(1)} + \delta\left(\frac{\partial\varphi^{(1)}}{\partial x}\right)_0 + \frac{\partial^2}{2!}\left(\frac{\partial^2\varphi^{(1)}}{\partial x^2}\right)_0 + \ldots$$

$$\varphi_1^{(4)} = \varphi_0^{(4)} + \delta\left(\frac{\partial\varphi^{(4)}}{\partial x}\right)_0 + \frac{\partial^2}{2!}\left(\frac{\partial^2\varphi^{(4)}}{\partial x^2}\right)_0 + \ldots$$

$$\varphi_3^{(2)} = \varphi_0^{(2)} - \delta\left(\frac{\partial\varphi^{(2)}}{\partial x}\right)_0 + \frac{\partial^2}{2!}\left(\frac{\partial^2\varphi^{(2)}}{\partial x^2}\right)_0 + \ldots$$

$$\varphi_3^{(3)} = \varphi_0^{(3)} - \delta\left(\frac{\partial\varphi^{(3)}}{\partial x}\right)_0 + \frac{\partial^2}{2!}\left(\frac{\partial^2\varphi^{(3)}}{\partial x^2}\right)_0 + \ldots$$

după axa Oy:

$$\varphi_2^{(1)} = \varphi_0^{(1)} + \delta\left(\frac{\partial\varphi^{(1)}}{\partial y}\right)_0 + \frac{\partial^2}{2!}\left(\frac{\partial^2\varphi^{(1)}}{\partial y^2}\right)_0 + \ldots$$

$$\varphi_2^{(2)} = \varphi_0^{(2)} + \delta\left(\frac{\partial\varphi^{(2)}}{\partial y}\right)_0 + \frac{\partial^2}{2!}\left(\frac{\partial^2\varphi^{(2)}}{\partial y^2}\right)_0 + \ldots$$

$$\varphi_4^{(3)} = \varphi_0^{(3)} - \delta\left(\frac{\partial\varphi^{(3)}}{\partial y}\right)_0 + \frac{\partial^2}{2!}\left(\frac{\partial^2\varphi^{(3)}}{\partial y^2}\right)_0 + \ldots$$

$$\varphi_4^{(4)} = \varphi_0^{(4)} - \delta\left(\frac{\partial\varphi^{(4)}}{\partial y}\right)_0 + \frac{\partial^2}{2!}\left(\frac{\partial^2\varphi^{(4)}}{\partial y^2}\right)_0 + \ldots$$

Prin sumare şi neglijând termenii proporționali cu δ^3 se obține:

$$h_0 = \frac{1}{2\sum_{i=1}^{4} k^{(i)}}\left[\left(k^{(1)} + k^{(2)}\right)h_1 + \left(k^{(1)} + k^{(2)}\right)h_2 + \left(k^{(2)} + k^{(3)}\right)h_3 + \left(k^{(3)} + k^{(4)}\right)h_4\right]$$

În cazul omogen $k^{(i)} = k$ se obține relația corespunzătoare mediilor omogene prezentată mai sus

Pentru nodurile marginale se deduc relații de formă adecvată condițiilor la limită şi formei conturului. În final toate relațiile formează un sistem liniar de ecuații de forma:

$$[k]\{h\} = \{R\}$$

unde:

$[k]$ = matricea coeficienților, matrice de ordinul n x n;

$\{h\}$ = matricea coloană a valorilor nodale a funcțiilor căutate;

$\{R\}$ = matricea coloană a termenilor liberi care se determină din condițiile la limită.

Metoda volumelor finite (celulelor finite) conține ecuații relativ similare, deosebirea constând în faptul că valorile "h_i" reprezintă înălțimile piezometrice calculate în centrul celulei (volumului).

3.2 Metoda elementelor finite.

Bazele metodei în varianta Ritz sunt oferite de formularea variațională echivalentă a problemei la limită [23], [25]. Soluția $h = h(x, y)$ se caută sub forma

$$h(x, y) = \sum_{i=1}^{n} c_i \cdot \phi_i(x, y)$$

unde:

- c_i sunt constante nedeterminate;
- $\phi_i(x, y)$ sunt funcții de interpolare date care satisfac condițiile la limită esențiale (Dirichlet):

$h|_{C_1} = h_0$

Soluția aproximativă înlocuită în expresia funcționalei variaționale definită în paragraful 1.4 devine:

$$U(h) = \frac{1}{2}\int_D \left[T\left(\sum c_i \frac{\partial \phi_i}{\partial x}\right)\cdot\left(\sum c_j \frac{\partial \phi_j}{\partial x}\right) + T\left(\sum c_i \frac{\partial \phi_i}{\partial y}\right)\cdot\left(\sum c_j \frac{\partial \phi_j}{\partial y}\right)\right] -$$
$$\int_D \varepsilon \cdot \sum c_i \phi_i dxdy + \int_{C_2} q \sum c_i \phi_i dl$$

În baza principiului variațional

$$\delta U = 0 \quad \Rightarrow \quad \frac{\partial U}{\partial c_i} = 0$$

se obține un sistem liniar de ecuații:

$[A]\cdot[C] = [B]$

unde:

A - reprezintă matricea coeficienților sistemului:

$$A_{j,i} = \int_D \left(T\frac{\partial \phi_i}{\partial x}\cdot\frac{\partial \phi_j}{\partial x} + T\frac{\partial \phi_i}{\partial y}\cdot\frac{\partial \phi_j}{\partial y}\right) dxdy$$

B - reprezintă matricea coloană a termenilor liberi având componentele:

$$B = \int_D \varepsilon \cdot \phi_i dxdy - \int_{C_2} q\phi_i^* dl$$

C- reprezintă matricea coloană a coeficienților nedeterminați

ϕ_i^* - sunt funcții de interpolare globale pe porțiunea C_2 a frontierei domeniului.

Metoda elementului finit constă în aplicarea procedeului de mai sus (Ritz), pentru o discretizare dată, pentru fiecare element în parte. În acest sens domeniul mişcării se împarte în elemente finite $\Delta^{(e)}(e = 1,2,3,...m)$ deci un număr m de elemente. Pe fiecare element în parte se poate considera valabilă o aproximare similară obținută mai sus pe domeniul D. Astfel pentru fiecare element finit în parte se obține un sistem local de ecuații având coeficienții:

$$A_{NM}^{(e)} = \int_{D^e} \left[T_x^e \frac{\partial \phi_N^e}{\partial x} \frac{\partial \phi_M^e}{\partial x} + T_y^e \frac{\partial \phi_N^e}{\partial y} \frac{\partial \phi_M^e}{\partial y} \right] ds^e$$

$$B_N^e = \int_{D^e} \varepsilon \cdot \phi_N^e ds^e + \int_{C^e} q^e \phi_n^e dl$$

În matricea coloană „C" locul constantelor c_i va lua valorile nodale „h_i^e" (fig. 3.3)

Aceste ecuații reprezintă ecuațiile standard pentru elementele finite.

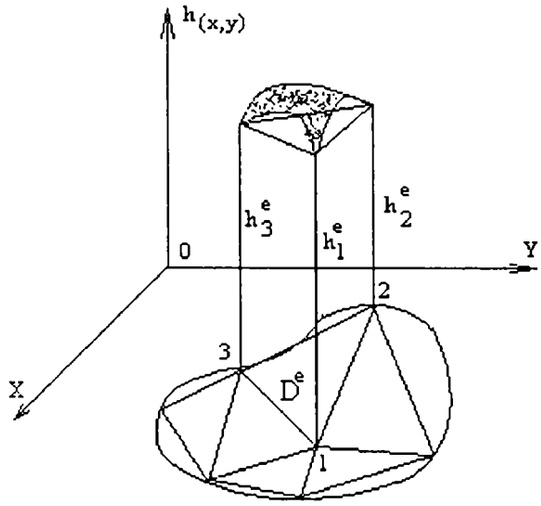


Fig. 3.3 Discretizarea domeniului (MEFIN)

Prin asamblarea sistemelor locale de tipul de mai sus se obține un sistem global de forma:

$$A_{j,i} \cdot h_i = B_i \quad ; \quad i = \overline{1,n}, \quad j = \overline{1,n}$$

unde se sumează în raport cu i după regula lui Einstein;

$$A_{J,I} = \sum_{e=1}^{m} A_{N,M}^{(e)} \Delta_{N,I}^{(e)} \Delta_{M,J}^{(e)}$$

$$B_I = \sum_{e=1}^{m} B_N^{(e)} \Delta_{N,I}^{(e)}$$

unde $\Delta_{N,I}^{(e)}$ sunt matricile Booleene;

$$\Delta_{N,I}^{(e)} = \begin{cases} 1 & \text{pentru } N = i \\ 0 & \text{pentru } N \neq i \end{cases}$$

i este indicele nodului global iar N este indicele elementului "e"

3.3 Metoda elementelor de frontieră

Această metodă în variantă indirectă [23], [25] se bazează pe reprezentarea integrală a soluției cu ajutorul soluției fundamentale:

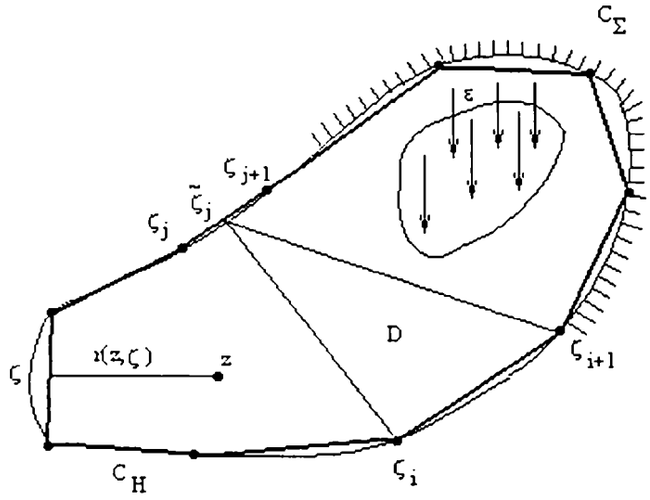
$$h(z) = -\frac{1}{2\pi T}\int_C \psi(\zeta)\ln r(z,\zeta)dl - \frac{1}{2\pi T}\int_{\tilde{D}} \varepsilon(\tilde{z})\ln r(z,\tilde{z})ds + C$$

pentru potențialul hidraulic (înălțime piezometrică), respectiv

$$U(z) = \frac{1}{2\pi}\int_D \psi(\zeta)\frac{\partial}{\partial n}\ln r(z,\zeta)dl + \frac{1}{2\pi}\int_{\tilde{D}} \varepsilon(\tilde{z})\frac{\partial}{\partial n}\ln r(z,\tilde{z})ds$$

pentru fluxul după direcția normală „n" în punctul $z \in D$

Pentru aplicarea practică a metodei se trece la limită $z \to \zeta_0 \in C_0$ obținându-se astfel ecuații integrale cu necunoscutele $\psi(\zeta)$. Pentru aceasta, se discretizează frontiera domeniului printr-un contur poligonal format din elementele de frontieră (fig.3.4.). Pe fiecare element se consideră o aproximare a distribuției ψ_i. Pe baza acestor aproximări va rezulta din ecuațiile integrale un sistem de ecuații liniare pentru determinarea distribuțiilor ψ_i şi a constantei C.



$C_\Sigma \cup C_H$ = frontiera domeniului

D = domeniul miscarii

Fig 3.4 Discretizarea domeniului (MEFRO)

În cazul mai simplu când $\varepsilon = 0$ adică nu există aport de debit pe suprafața domeniului se obține un sistem liniar de forma:

$$\begin{cases} \sum_{i=1}^{n} a_{j,i} \frac{\psi_i}{T} + C = h_{o,j} \\ \sum_{\substack{i=1 \\ i \neq j}}^{n} b_{j,i} \frac{\psi_i}{T} - \frac{1}{2} \frac{\psi_j}{T} = 0 \\ \sum_{i=1}^{n} l_{i,i+1} \frac{\psi_i}{T} + \frac{\varepsilon}{T} S = 0 \end{cases}$$

existența unor obiecte interioare puțuri, drenuri, etc. va complica şi mai mult ecuațiile. Reprezentarea lor se va face în capitolul V.

Coeficienții $a_{j,i}$ se calculează pe baza expresiilor:

$$a_{j,i} = \frac{1}{2\pi} \int_{\xi_i}^{\xi_{i+1}} \ln r\left(\tilde{\zeta}_j, \zeta\right) dl$$

$$b_{j,i} = \frac{1}{2\pi} \int_{\xi_i}^{\xi_{i+1}} \frac{\partial \ln r\left(\tilde{\zeta}, \zeta\right)}{\partial n} dl$$

unde $\tilde{\zeta}_j$ este punctul mijlociu al elementului $e_{j,j+1}$.

În urma rezolvării sistemului se determină valorile lui ψ_i care introduse în relații h, respectiv u permit obținerea înălțimii piezometrice h şi a fluxului după direcția „n" într-un punct M din interiorul domeniului $(M \in D)$.

$$h(M) = \sum_{i=1}^{n} a_{M,i} \frac{\psi_i}{T} + \frac{\varepsilon}{T} \sum_{\nu=1}^{m} c_{(M,\nu)} + C$$

$$u(M) = \sum_{i=1}^{n} b_{M,i} \frac{\psi_i}{T} - \frac{\varepsilon}{T} \sum_{\nu=1}^{m} d_{(M,\nu)}$$

Dacă $\varepsilon \neq 0$, adică există un aport de debit, atunci este necesară discretizarea subdomeniului pe care $\varepsilon \neq 0$.

Dacă $\varepsilon = const$ atunci este suficientă discretizarea numai a frontierei. În acest caz sistemul de ecuații liniare este similar cu cel de mai sus cu mențiunea că mai apare termenul corespunzător lui ε.

Existența unor obiecte interioare puțuri, drenuri etc. va complica şi mai mult ecuațiile. Reprezentarea lor se va face în capitolul V.

CAPITOLUL IV

ANALIZA ERORILOR METODELOR NUMERICE ÎN REPREZENTAREA SINGULARITĂȚILOR SPECIFICE CAPTĂRILOR SUBTERANE

Reprezentarea corectă a puțurilor şi drenurilor de lungime finită, adică introducerea comportamentelor singulare specifice în general captărilor subterane, constituie problema importantă în elaborarea modelelor numerice în domeniul hidraulicii subterane.

În cazul în care rezolvarea problemelor de hidraulică subterană prin metode analitice devine dificilă din cauza complexității problemelor, de exemplu cazul domeniilor cu o geometrie complicată, cazul mediilor neomogene etc. se recurge la modelarea numerică a mişcării apelor subterane.

4.1 Metoda diferențelor finite.

Metoda diferențelor finite reprezintă o metodă de modelare numerică bazată pe discretizarea întregului domeniu al mişcării. Această metodă elimină dificultățile legate de geometria complexă a domeniului, inclusiv cele legate de modul de dispunere al puțurilor sau al drenurilor.

Principalele probleme care apar la modelele numerice cu discretizarea domeniului constau în dificultatea reprezentării corecte a dependenței dintre denivelări şi debite sau distribuții de debite aceasta mai ales în zone sau vecinătăți apropiate puțurilor sau drenurilor unde expresia înălțimii piezometrice are din punct de vedere matematic un caracter singular (ex. logaritmic).

Pentru analizarea şi studiul acestor efecte în cazul bidimensional s-a utilizat programul A.S.M. (Aquifer Simulation Modell) [41], [42] care are ca bază de calcul metoda diferențelor finite (varianta celulară). A.S.M. a fost dezvoltat ca şi mijloc de studiu în domeniul hidraulicii subterane şi al ingineriei mediului. Programul modelează mişcarea apei subterane precum şi fenomene de transport luând în

considerare domenii bidimensionale ale mişcării. Pentru discretizarea domeniului se foloseşte o reţea de celule dreptunghiulare. Reprezentarea puţului se realizează prin introducerea unui debit concentrat în acele celule (mijlocul celulei) în care sunt plasate puţurile. În urma rulării programului rezultatele sunt: câmpul hidroizohipselor şi denivelarea în puţ. Ceea ce este important de menţionat este faptul că denivelarea astfel calculată, este afectată de erori considerabile şi acesta deoarece metoda nu permite luarea în considerare a razei puţului ca parametru. Exceptând vecinătatea singularităţilor (puţuri, linii de drenaj, etc.) metoda de calcul oferă în general rezultate foarte bune.

În literatura de specialitate sunt prezentate diferite posibilităţi pentru rezolvarea problemelor menţionate mai sus, adică reprezentarea cât mai corectă a potenţialelor în vecinătatea singularităţilor.

În principal aceste soluţii propun următoarele metode:

- îndesirea reţelei de discretizare a domeniului în zonele în care se urmăreşte obţinerea de rezultate cu o acurateţe cât mai mare;
- folosirea unor funcţii de interpolare de grad superior în cadrul algoritmului de calcul;
- amplasarea singularităţilor în interiorul unui element şi folosirea suprapunerii efectelor.

O lucrare semnificativă din punct de vedere al tratării problemelor legate de erorile care apar în cadrul metodelor numerice cu diferenţe finite aparţine lui Klenke [43]. Ea tratează doar cazul unui puţ singular cu debit dat, evaluând deci eroarea denivelării.

David I. [24] a continuat aceste cercetări referitoare la evaluarea erorilor care apar în cazul modelelor numerice de simulare pentru puţuri şi drenuri de lungime finită în ambele variante de funcţionare (Q, H).

4.1.1 Evaluarea erorilor metodei diferenţelor finite pentru puţuri perfecte, drenuri de lungime finită şi puţuri parţial penetrate.

Cazul puţurilor perfecte.

Se prezintă câteva rezultate care generalizează pe cele stabilite de Klenke[43]. Pentru evaluarea erorilor respectiv calculul erorii "ε_h" al denivelării în puţ în cazul când debitul este dat s-a stabilit următoarea relaţie de calcul (I. David [24]);

$$\varepsilon_h = \frac{|\Delta H_A - \Delta H_N|}{\Delta H_A} = \left| 1 - \frac{2\sum_{i=1}^{n} \frac{1}{2(r/\Delta r + 2i - 1)}}{\ln \frac{R}{r}} \right|$$

$$n = (R - r)/\Delta r$$

unde semnificația mărimilor este următoarea:

- ΔH_A reprezintă denivelarea calculată analitic;
- ΔH_N reprezintă denivelarea calculată numeric;
- Δr reprezintă pasul de discretizare al rețelei;
- R raza de influență;
- r distanța la care se calculează denivelarea.

Valorile calculate ale erorii pentru denivelarea generată de puț la diferite distanțe de acesta este prezentată în tabelul 1.

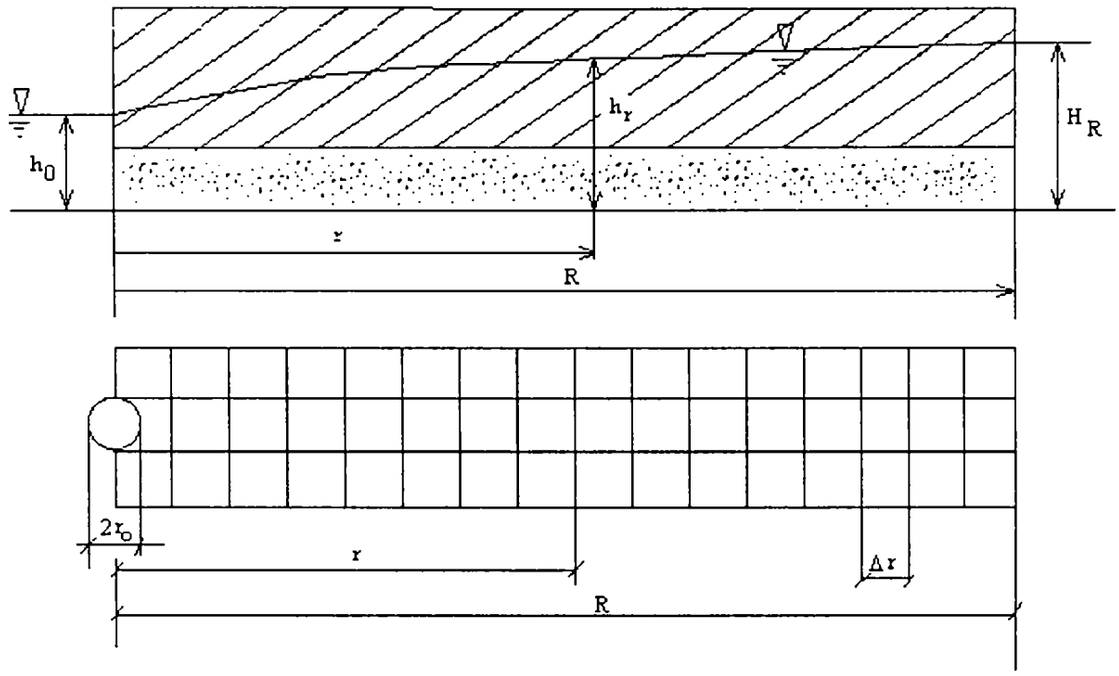


Fig.4.1 Rețeaua de discretizare în cazul puțului.

Valorile erorii de calcul a denivelării "ε_h " Tabelul 1

		Distanța față de puț r(m)					
		0.1	1	10	100	200	500
Pasul rețelei Δr (m)	50	46.20	28.20	8.20			
	100	53.70	38.20	16.30	1.52		
	200	61.20	48.30	27.10		2.16	
	500	71.10	61.50	44.00			3.82

Observație:

Coloanele tabelului reprezintă eroarea exprimată în procente a denivelării totale, în cazul unui puț cu raza $r_0 = 0{,}1m$ şi raza de influență $R = 1000m$.

Pentru calculul erorii debitului unui puț "ε_Q " în cazul unei denivelări date s-a stabilit relația (I. David [24]):

$$\varepsilon_Q = \frac{|\Delta Q_A - \Delta Q_N|}{\Delta Q_A} = \left| 1 - \frac{\ln \frac{R}{r_0}}{2\sum_{i=1}^{n} \frac{1}{2\frac{r_0}{\Delta r} + 2i - 1}} \right|$$

unde semnificația mărimilor este următoarea:

- Q_A reprezintă debitul puțului calculat analitic;
- Q_N reprezintă debitul puțului calculat numeric;
- R raza de influență;
- r_0 raza puțului;
- r raza la care se calculează valoarea debitului.

Rezultatele obținute în urma calculelor pentru 3 valori ale razei puțului şi aceeaşi rază de influență $R = 1000m$ sunt prezentate în tabelul 2.

Eroarea relativă a debitului unui puț "ε_Q " Tabelul 2

		Pasul de discretizare al rețelei (m)					
		10	25	50	100	200	500
Raza puțului (m)	0.10	44.60	64.80	87.00	115.00	157.00	245.00
	0.25	30.20	48.50	68.40	94.10	131.40	211.00
	1.00	8.50	23.60	40.20	61.60	92.70	159.00

Rezultatele obținute pun în evidență erori de calcul semnificative mai ales pentru un pas de discretizare relativ mare. În ceea ce priveşte denivelarea (tabelul 1) erorile sunt mari în puț şi scad rapid, devenind semnificative la distanțe relativ mici (circa 10m) de puț, rezultat cunoscut de altfel. (ex. Klenke [43]).

Erorile sunt mult mai semnificative cu cazul calculului numeric al debitului (tabelul 2) pentru o denivelare dată. În practică apar asemenea situații la calculul epuismentelor când se cere determinarea debitului extras pentru a realiza o anumită denivelare prescrisă sau în cazul puțurilor de infiltrare pentru îmbogățirea acviferului, caz în care nivelul apei în puț este limitat de suprafața terenului. În ambele cazuri debitul calculat numeric, mai ales în cazul puțurilor cu rază obişnuită, mică şi medie (de 0,1-0,25m) este afectat de erori semnificative de 30-45% chiar la o discretizare relativ fină $\Delta r = 10m$.

Cazul drenului.

Pentru evaluarea erorii denivelării totale într-un dren perfect de lungime finită "ε_H " se poate folosi formula (I. David [24]):

$$\varepsilon_H = \frac{|\Delta H_A - \Delta H_N|}{\Delta H_A} = \left| 1 - \frac{\sum_{i=1}^{n} \frac{2}{\sqrt{(2\frac{l}{\Delta r} + 2i - 1)^2 - 4\frac{l^2}{\Delta r^2}}}}{ln(\frac{R}{l} + \sqrt{\frac{R^2}{l^2} - 1})} \right|$$

unde semnificația mărimilor este următoarea:

- ΔH_A reprezintă denivelarea calculată analitic;
- ΔH_N reprezintă denivelarea calculată numeric;
- l lungimea drenului;
- R raza de influență.

Rezultatele erorii denivelării totale pentru un dren perfect cu lungime de $2l$=100, 200, 400m şi rază de influență de 1000m sunt prezentate în tabelul 3.

a)

b)

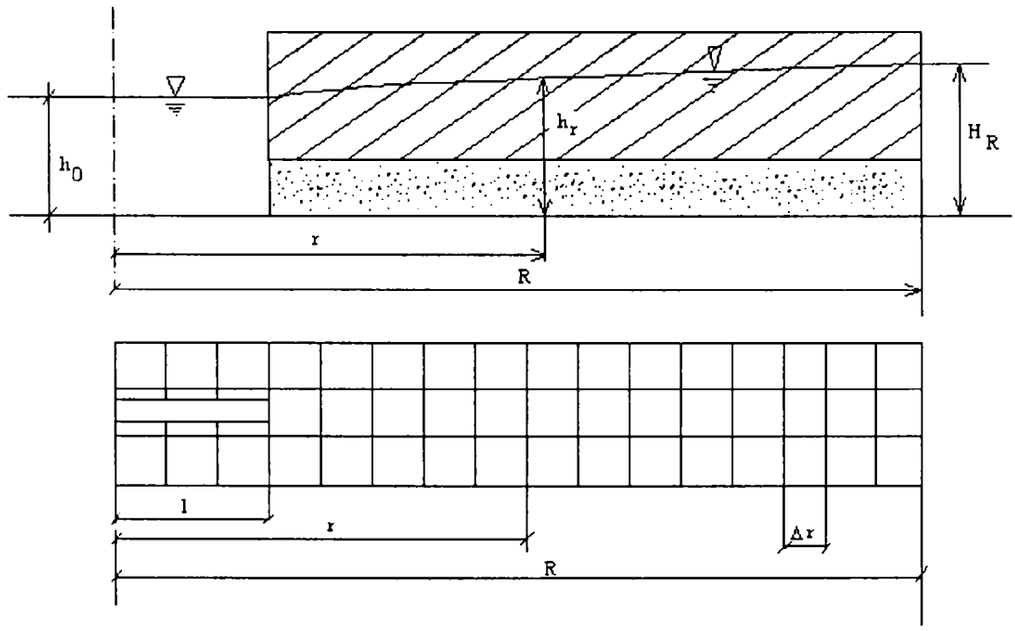


Fig. 4.2 Rețeaua de discretizare în cazul drenului

a) secțiune în plan vertical , b) vedere în plan orizontal

Eroarea relativă a denivelării ε_H[%] pentru. un dren de lungime 2l şi R=1000m

Tabelul 3

		Pasul reţelei de discretizare (m)			
		25	50	100	200
Lung. Dren (m)	100	8.30	11.90	-	-
	200	7.20	10.20	13.3	-
	400	6.60	9.40	15.7	19.00

Din cauza distribuţiei neuniforme de debit în lungul liniei de dren nu s-a putut stabili o formulă pentru estimarea erorii debitului. În cazul drenurilor se observă că erorile denivelării totale sunt în general mai mici decât cele corespunzătoare puţurilor (tabelul 3). În ceea ce priveşte debitul s-au constatat erori foarte mari la distribuţia acestuia de-a lungul drenului, la capetele drenului se ajunge la erori de până la 50-60%.

Cazul puţurilor parţial penetrate.

Un caz deosebit de captare subterană îl reprezintă puţurile imperfecte din cauza penetrării parţiale de natură tridimensională.

În [28] se analizează calculul hidraulic al acestor tipuri de captări prezentându-se dificultăţile datorită caracteristicilor tridimensionale ale mişcării apei subterane în vecinătatea puţului. În acest sens se arată că chiar şi în cazul mediilor omogene rezolvarea ecuaţiei generale a mişcării apei subterane

$$\frac{\partial^2 h}{\partial x^2}+\frac{\partial^2 h}{\partial y^2}+\frac{\partial^2 h}{\partial z^2}=0$$

în care înălţimea piezometrică $h(x,y,z)$ definită în domeniul D al mişcării trebuie să satisfacă condiţii la limită destul de complexe.

Domeniul mişcării D este definit prin relaţia:

$$D=\left\{(x,y,z)\middle| x^2+y^2<R, 0<z<m\right\}-\left\{(x,y,z)\middle| x^2+y^2\le r_0^2; a\le z<m\right\}$$

Funcţia de potenţial $h(x,y,z)$ trebuie să satisfacă următoarele condiţii la limită:

- pe frontiera domeniului;

$$h_{|\Gamma_R}=H_R \quad , \quad \Gamma_R=\left\{(x,y,z)\middle| x^2+y^2=R^2, 0\le z\le m\right\}$$

- pe frontiera puțului;

$$h_{|\Gamma_P} = H_0 \quad , \quad \Gamma_P = \left\{(x,y,z)\middle| x^2 + y^2 = r_0^2, a \le z \le m\right\}$$

- pe suprafețele impermeabile care delimitează domeniul în partea inferioară (S_i) și superioară (S_s)

$$\frac{\partial h}{\partial n}_{|S_i \cup S_s} = 0 \quad , \quad S_i = \left\{(x,y,z)\middle| x^2 + y^2 < R^2, z = 0\right\}$$

$$S_s = \left\{(x,y,z)\middle| r_0^2 < x^2 + y^2 < R^2, z = m\right\}$$

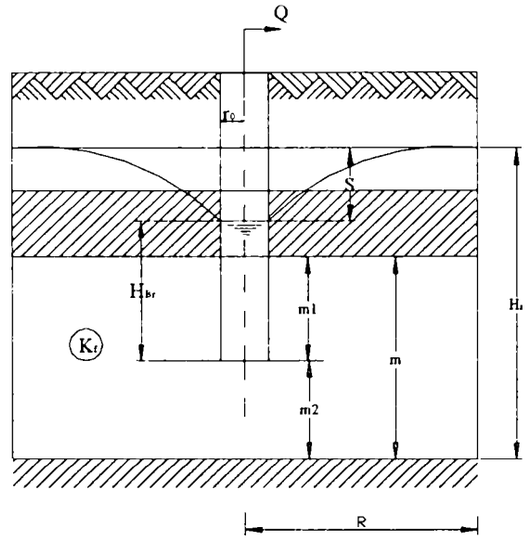


Fig 4.3 Schema puțului imperfect

Soluțiile analitice clasice [48], [39] au fost obținute prin simplificarea condițiilor la limită care trebuiesc respectate de către funcția potențial $h(x,y,z)$ pe puț în ipoteza că $r_0 << R$. În aceste condiții care în general acoperă cazurile reale, puțul poate fi înlocuit cu o sursă liniară distribuită de-a lungul axului puțului. Luând în considerare aceste condiții, Muskat [48] a obținut o formulă teoretică de calcul a debitului pentru puțul imperfect care are următoarea formă:

$$\frac{Q}{k_f m} = \frac{2\pi(H_R - H_0)}{ln\frac{R}{r_0} + \left(\frac{m}{l} - 1\right) ln\frac{4m}{r_0} - \frac{m}{2l} f\left(\frac{l}{m}\right)}$$

unde valorile funcției f sunt prezentate în tabelul nr. 4

Tabelul 4

L/m	0.05	0.08	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f(l/m)	8	7	6.5	5.2	4.2	3.6	3	2.4	1.9	1.4	0.8	0

Relația lui Muskat permite determinarea debitului unui puț parțial penetrat în cazul când nu se cunoaște pierderea de sarcină $H_R - H_P$ împreună cu alți parametrii, geometrici prezentați în figura 4.3

Totodată relația poate fi folosită şi la determinarea denivelării în cazul în care se cunoaşte debitul puțului parțial penetrat.

Un studiu al erorilor în modelarea numerică a puțurilor imperfecte a fost făcut de către I. David şi Gaude P. [29]. Pentru modelare s-a folosit programul MODFLOW, program de modelare numerică destinat aplicațiilor în domeniul hidraulicii subterane şi a transportului poluanților răspândit mai ales în S.U.A. Acest program a fost realizat şi dezvoltat în anul 1994 în Canada şi are la ora actuală peste 3500 de utilizatori care îşi desfăşoară activitatea în aceste domenii specifice [59]. Programul foloseşte metoda volumelor finite pentru a realiza o discretizare tridimensională a domeniului.

Analiza erorilor care apar la modelarea numerică a puțurilor imperfecte s-a efectuat pentru un puț imperfect aflat într-un strat acvifer omogen luând în considerare două variante:

- cazul în care se cunoaşte valoarea debitului puțului imperfect şi se determină înălțimea piezometrică din interiorul puțului;

- caz în care se cunoaşte înălțimea piezometrică în interiorul puțului şi pe conturul domeniului, deci pierderea de sarcină şi se determină debitul puțului imperfect.

Elementele caracteristice ale modelului sunt:

- raza puțului imperfect $r_0 = 0{,}5m$;
- raza de influență a puțului imperfect $R = 100m$;
- coeficientul de conductivitate hidraulică $k_f = 0{,}000278m/s$;
- grosimea stratului acvifer $m = 20m$.
- $H_R = 24m$, înălțimea piezometrică la distanța razei de influență

Pentru a putea ilustra cât mai corect influența discretizării tridimensionale a modelului în vecinătatea puțului imperfect s-au efectuat un experiment numeric sistematic considerând următoarele tipuri de discretizare:

- o rețea de discretizare în plan orizontal de $10x10m^2$, $5x5m^2$, $2x2m^2$ şi $1x1m^2$ combinată cu trei tipuri diferite de discretizare efectuate în plan vertical fiecare cuprinzând 5, 10 şi respectiv 15 straturi.

Evaluarea erorilor s-a făcut prin compararea rezultatelor numerice cu cele analitice calculate cu formula lui Muskat.

Rezultatele comparative pentru varianta 1 respectiv varianta 2 de calcul sunt prezentate în tabelul 5 şi tabelul 6

Varianta 1 Tabelul 5

Valori calculate ale înălțimii piezometrice în puț şi ale erorii relative între metoda numerică şi soluția analitică (Q dat)								
	4m	$\varepsilon_{H[\%]}^{4m}$	8m	$\varepsilon_{H[\%]}^{8m}$	12m	$\varepsilon_{H[\%]}^{12m}$	16m	$\varepsilon_{H[\%]}^{16m}$
Muskat	22,0		22,0		22,0		22,0	
MODFLOW								
10x10,5	22.9161	45.81	22.6637	33.18	22.5578	27.89	22.4998	24.99
10x10,10	22.9085	45.43	22.6586	32.93	22.5551	27.76	22.4986	24.93
10x10,15	22.9069	45.35	22.6575	32.87	22.5545	27.73	22.4983	24.92
5x5,5	22.3725	18.62	22.2025	10.13	22.1774	8.87	22.1756	8.78
5x5,10	22.3470	17.35	22.1923	9.61	22.1737	8.69	22.1743	8.71
5x5,15	22.3424	17.12	22.1911	9.55	22.1738	8.96	22.1747	8.74
2x2,5	21.5897	-20.52	21.6278	-18.61	21.7098	-14.51	21.7699	-11.51
2x2,10	21.5405	-22.97	21.6178	-19.11	21.7068	-14.66	21.7689	-11.56
2x2,15	21.5313	-23.43	21.6173	-19.14	21.7074	-14.63	21.7698	-11.51
1x1,5	21.0145	-49.27	21.2101	-39.50	21.3617	-31.92	21.4641	-26.80
1x1,10	20.9642	-51.79	21.2011	-39.95	21.3590	-32.05	21.4631	-26.85
1x1,15	20.9560	-52.20	21.2009	-39.96	21.3598	-32.01	21.4642	-26.79

Varianta 2 Tabelul 6

Valori calculate ale debitului în puț şi ale erorii relative între metoda numerică şi soluția analitică ($Ho=22m$)								
	4m	$\varepsilon_{Q[\%]}^{4m}$	8m	$\varepsilon_{Q[\%]}^{8m}$	12m	$\varepsilon_{Q[\%]}^{12m}$	16m	$\varepsilon_{Q[\%]}^{16m}$
Muskat	479,1	-	717,7	-	903,5	-	1060,5	-
MODFLOW								
10x10,5	1176.3	145.5	1499.3	108.9	1723.4	90.7	1866.7	76.0
10x10,10	1192.2	148.8	1510.7	110.5	1731.6	91.7	1871.3	76.5
10x10,15	1195.7	149.6	1513.2	110.8	1733.5	91.9	1872.5	76.6
5x5,5	812.9	67.7	1117.2	55.7	1342.3	48.6	1500.0	41.4
5x5,10	836.4	74.6	1134.5	58.1	1355.2	50.0	1508.4	42.2
5x5,15	841.5	75.6	1138.2	58.6	1358.1	50.3	1510.1	42.4
2x2,5	528.6	10.3	799.1	11.3	1011.8	12.0	1176.6	10.9
2x2,10	552.7	15.4	817.8	13.9	1026.6	13.6	1186.2	11.9
2x2,15	560.6	17.0	823.9	14.8	1031.6	14.2	1190.8	12.3
1x1,5	406.9	-15.1	647.9	-9.7	845.9	-6.4	1003.9	-5.3
1x1,10	431.5	-9.9	665.8	-7.2	857.3	-5.1	1014.4	-4.3
1x1,15	429.9	-10.2	668.7	-6.8	862.4	-4.5	1025.0	-3.3

4.1.2 Concluzii şi comentarii cu privire la erorile introduse de metoda diferenţelor finite.

- pentru **cazul puţului perfect** cu debit dat se observă că eroarea denivelării totale este mare chiar şi pentru o reţea cu pas fin de discretizare, ea scăzând însă pe măsură ce ne îndepărtăm de puţ rămânând semnificativă şi la distanţe mari în cazul în care reţeaua de discretizare are un pas mare;
- erorile relative ale debitului **puţului perfect** în cazul unei denivelări totale date sunt mult mai mari faţă de cazul puţului cu debit dat. Chiar şi în cazul unui puţ cu diametru de 0,5m. şi un pas de discretizare al reţelei de 10m. eroarea calculată depăşeşte 30%;
- în cazul **drenurilor de lungime finită** erorile denivelării totale în dren sunt în general mai mici decât cele corespunzătoare puţurilor. În ceea ce priveşte debitul s-a constatat erori mari la distribuţia acestuia în lungul liniei de dren mai ales la capetele drenului.

În cazul **puţurilor imperfecte** ţinând cont de rezultatele prezentate în cele două variante de rulare s-au desprins următoarele concluzii:

Pentru varianta 1;

- mărirea gradului de discretizare în plan orizontal nu conduce la reducerea erorilor relative $\varepsilon_{H[\%]}$ între soluţia numerică şi cea analitică. După cum se observă nu se poate recomanda o soluţie care să ofere nişte rezultate afectate de erori $\varepsilon_{H[\%]}$ care să se apropie de zero;
- discretizare verticală mai accentuată are doar o relevanţă redusă în ceea ce priveşte $\varepsilon_{H[\%]}$. Mărirea gradului de discretizare în plan vertical poate conduce la $\varepsilon_{H[\%]}$ chiar mai mari decât în cazul unei discretizări reduse;
- creşterea lungimii puţului imperfect de la 4m la 16m reduce eroarea $\varepsilon_{H[\%]}$ la jumătate.

Pentru varianta 2;

- discretizarea în plan orizontal are o mare influenţă asupra erorii relative $\varepsilon_{Q[\%]}$ între soluţia numerică şi cea analitică. Această eroare relativă, după cum se observă variază între 45 şi –53%, şi nu poate fi redusă nici în cazul unei discretizări mai mari;
- diferitele grade de discretizare în plan vertical nu au o influenţă semnificativă asupra $\varepsilon_{Q[\%]}$;
- cu cât gradul de penetrare a puţului creşte erorile relative se reduc, ceea ce ne conduce la concluzia că cu cât ne apropiem de situaţia unui puţ perfect erorile relative $\varepsilon_{Q[\%]}$ se reduc, rămânând însă semnificative.

Aceste concluzii şi comentarii prezentate pe larg în [29] stabilesc clar că metodele cu diferenţe finite (volume finite) pot conduce la erori semnificative în situaţiile în care mişcarea prezintă singularităţi bi sau tridimensionale aşa cum sunt în general captările subterane de orice tip

Analiza erorilor singularităților în cazul aplicării metodei elementelor finite s-a realizat prin intermediul programului AQUA [57] dezvoltat de către firma Vatnaskil Consulting Engineering. Acest program permite posibilități de rezolvare a problemelor de hidraulică subterană, probleme de transport etc.

Problema analizată a fost şi în acest caz dependența rezultatelor modelării captărilor subterane (Q,H) funcție de gradul de discretizare al domeniului mişcării. În acest sens I.David, G. Eleş [27] au efectuat o serie de calcule sistematice în cazuri specifice pentru puțuri şi drenuri.

4.2.1 Evaluarea erorilor folosind metoda elementelor finite pentru puțuri şi drenuri de lungime finită.

Cazul unui puț izolat alimentat pe contur, denivelare constantă dată.

Pentru studiul acestei probleme s-a avut în vedere păstrarea unei denivelări constante şi rafinarea rețelei de elemente finite dinspre cercul exterior de alimentare C_E de rază R, înspre conturul puțului C_0 de rază r. Razele cercurilor interioare pe care au fost dispuse nodurile discretizării au fost calculate cu următoarea relație:

$$R_{i-1} = \frac{R_i}{2}$$

S-a obținut astfel un număr de 10 cercuri cu mărimea razelor prezentată în tabelul 8, incluzând cercul exterior C_E şi C_o prin care defineşte puțul în plan orizontal.

Pe conturul fiecărui cerc au fost calculate coordonatele geometrice a 12 noduri care au fost poziționate pe conturul fiecărui cerc (fig.4.4 şi tab.7). În tabelul 9 pentru fiecare variantă se indică numărul nodurilor luate în calcul, numărul elementelor, tipul variantei de calcul şi eroarea de calcul determinată.

Pentru acest caz s-a avut în vedere evaluarea valorică a erorii în ceea ce priveşte debitul total al puțului calculat prin metoda elementelor finite cu programul AQUA față de debitul real al puțului calculat prin metoda analitică pentru o denivelare constantă (formula cunoscută a puțurilor). Rezultatele obținute sunt prezentate în tabelul 9.

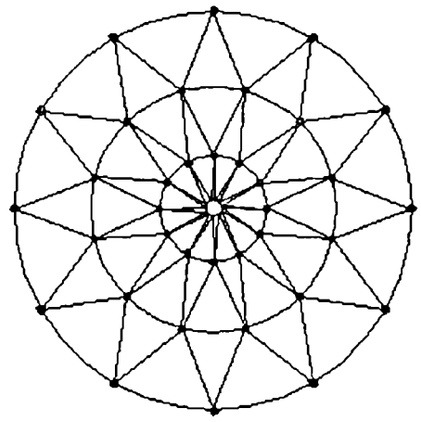


Fig. 4.4 Modul de discretizare a domeniului în cazul unui puț

Tabelul 7

Var.	Modul de distribuire al cercurilor concentrice									
V_0	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
V_1	C_0		C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
V_2	C_0			C_3	C_4	C_5	C_6	C_7	C_8	C_9
V_3	C_0				C_4	C_5	C_6	C_7	C_8	C_9
V_4	C_0					C_5	C_6	C_7	C_8	C_9
V_5	C_0		C_2		C_4		C_6		C_8	C_9
V_i	Varianta cu rețeaua îndesită									

Semnificațiile notațiilor folosite în tabelul 7 sunt prezentate în cadrul tabelului 8:

Tabelul 8

Cerc	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
R [m]	0.25	0.5	0.78	1.56	3.12	6.25	12.50	25.00	50.00	100

Eroarea de calcul a debitului unui puț alimentat pe contur (deniv. const.) Tabelul 9

Varianta	Număr de elemente	Număr de noduri	Eroarea de calcul a debitului ε%
V_0	216	120	6.20%
V_1	192	108	7.61%
V_2	96	168	12.57%
V_3	84	144	21.83%
V_4	72	120	36.04%
V_5	72	120	15.57%
V_i	456	864	2.29%

Pentru calculul analitic s-a utilizat formula cunoscută:

$$Q = \frac{2\pi km(H - h_0)}{\ln \frac{R_0}{r_0}}$$

unde semnificația mărimilor care intervin este următoarea:

Q - debitul;
m - grosimea stratului acvifer;
k – coeficientul de filtrație;
H - potențialul pe conturul exterior;
h_0 - potențialul pe conturul puțului;
R_0 - raza domeniului;
r_0 - raza puțului.

Se constată aşa cum era de aşteptat că erorile sunt în general mai mici dacă pasul rețelei de elemente finite scade în vecinătatea puțului. Totuşi pentru o discretizare de circa 6m în vecinătatea puțului (varianta V_4) eroarea este de peste 36%, deci semnificativă.

Cazul unui puț izolat din care se extrage un debit constant.

În cazul acestei probleme s-a avut în vedere extragerea unui debit dat Q, concentrat într-un singur nod şi calculul denivelării corespunzătoare. În acest caz la aplicarea metodei elementelor finite nu mai este necesară precizarea diametrului puțului ceea ce face ca rezultatul calculului deci denivelarea obținută prin metoda elementelor finite să nu aibă nici o legătură cu dimensiunile geometrice ale puțului. Valoarea reală însă a denivelării depinde de caracteristicile geometrice ale puțului, respectiv de raza, diametrul acestuia.

În tabelul 10 se prezintă modul concentric de dispunere a cercurilor în cazul celor 4 variate luate în considerare.

În aceste condiții la un debit extras Q, impus (cunoscut) corespunde pe baza formulei puțului un număr arbitrar de denivelări în puț funcție de diametrul acestuia. În tabelul 11 sunt prezentate rezultatele de sinteză pentru patru variante de discretizare a domeniului şi constau în reprezentarea erorii metodei elementelor finite în cazul celor patru variante de discretizare.

Tabelul 10

Var	Modul de dispunere al cercurilor concentrice						
V_1	C_1	C_2	C_3	C_4	C_5	C_6	C_7
V_2	-	C_2	C_7	C_4	C_5	C_6	C_7
V_3	-	-	C_7	C_4	C_5	C_6	C_7
V_4	-	-	-	C_4	C_5	C_6	C_7

Tabelul 11

Varianta	Nr. de elemente	Nr. de noduri	Eroarea denivelării calculată pentru:	
			R_0/r_0=400	R_0/r_0=1000
V_1	85	186	2,20%	15,11%
V_2	73	132	13,20%	24,66%
V_3	61	108	24,00%	34,03%
V_4	49	87	34,80%	43,41%

Se constată şi în acest caz o variabilitate a erorilor destul de mare.

Cazul unei linii de drenaj de lungime finită.

Pentru evaluarea erorilor în cazul unei linii de drenaj de lungime finită s-a luat în considerare un dren perfect cu o lungime de 25 m. considerând cazurile de discretizare ale reţelei de elemente finite în cinci variante prezentate în cele ce urmează.

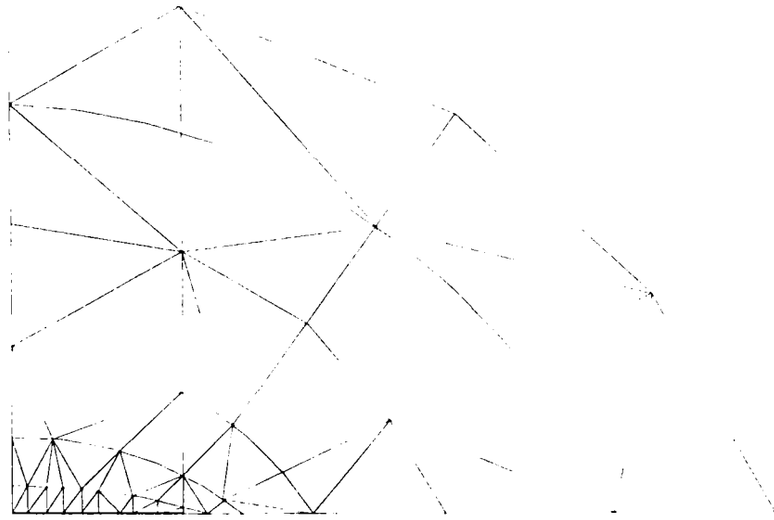


Fig. 4.5 Modul de discretizare al domeniului în cazul unei linii de drenaj.

În cadrul calculelor s-a avut în vedere păstrarea unui potenţial impus de 15 m pe conturul de alimentare şi un potenţial constant de 10m. în lungul liniei de dren. Discretizarea domeniului mişcării s-a efectuat sub forma unor elipse conform figurii 4.5 Rezultatul final este concretizat în evaluarea erorii în ceea ce priveşte debitul total al liniei de dren calculat prin metoda elementelor finite (programul AQUA) şi debitul liniei de dren real calculat prin metoda analitică. Aceste rezultate sunt prezentate în tabelul 12.

Tabelul 12

Nr. variantă	Nr. noduri	Nr. elemente	Eroare	Q_{calc}
V_1	38	53	5,03%	3,9980
V_2	43	61	4,21%	3,9666
V_3	54	81	3,52%	3,9405
V_4	95	151	2,44%	3,8993
V_i	340	604	0,62%	3,8301

Aşa cum s-a constatat şi la metoda diferenţelor finite, debitul total nu este afectat de erori semnificative.

4.2.2 Concluzii cu privire la erorile metodei elementelor finite.

În baza rezultatelor de mai sus se desprind o serie de concluzii dintre care se menţionează următoarele :

- în cazul unui puţ izolat alimentat simetric se remarcă faptul că pentru a obţine o precizie de sub 3% a debitului pentru o denivelare constantă, gradul de rafinare al reţelei trebuie să fie exagerat de mare şi cu precădere el trebuie să fie concentrat în vecinătatea puţului. Se observă că prin îndesirea reţelei varianta Vi, variantă prin care programul de calcul măreşte automat numărul de noduri şi de elemente într-o distribuţie oarecare se obţine o precizie de calcul foarte bună dar cu un număr exagerat de elemente, peste 600. Ţinând cont de acest aspect, pentru un sistem de captare de 50 de puţuri ar trebui să avem un număr de 30.000 de elemente;
- pentru un număr relativ mare de elemente şi noduri (192 elemente şi 108 noduri) se constată faptul că reducerea doar a primului cerc situat la o distanţă faţă de centrul puţului egală cu dublul razei conduce la mărirea erorii până la valoarea de 7,61%.
- ca şi concluzie finală se constată faptul că este foarte important menţinerea cercurilor din imediata vecinătate a puţului pentru a obţine aceleaşi valori ale erorii la acelaşi număr de elemente. Această concluzie poate fi uşor observată din compararea variantelor V_4 şi V_5 de calcul.
- în cazul unui puţ izolat din care se extrage un debit constant considerând două variante ale puţului una cu raza de 0,25m, respectiv 0,10m, se pot constata erori de calcul cuprinse între 2% şi 43%. Şi în acest caz valorile mici ale erorilor corespund unei reţele dense ca şi în exemplul precedent.
- în cazul liniei de dren, faţă de puţuri se observă că pentru obţinerea unei precizii de calcul sub 3% este suficientă alcătuirea unei reţele de discretizare formată dintr-un număr de 95 de noduri respectiv 151 de elemente. Această discretizare a condus în cele din urmă la o eroare de 2,44% ceea ce denotă o precizie de calcul bună.
- la drenuri, rămâne însă deschisă problema distribuţiei debitului de-a lungul drenului. La capetele drenului unde mişcarea are un caracter singular, diferenţele locale deci şi erorile sunt de ordinul de mărime constatat şi comentat în cazul puţurilor.

4.3 Evaluarea erorilor metodei elementelor de frontieră pentru puțuri .

Studiul erorilor în vecinătatea singularităților cu metoda elementelor de frontieră s-a realizat prin intermediul programului MEFRO creat și dezvoltat de către I.David și G.Eleș. Programul sursă a fost realizat în limbajul de programare PASCAL și va fi prezentat într-o variantă mai extinsă în cadrul capitolului VI.

Cazul puțurilor perfecte într-un acvifer omogen radial simetric.

În baza celor prezentate în paragrafele anterioare rezultă că atât metoda diferențelor finite, cât și metoda elementelor finite aproximează ecuațiile care guvernează mișcarea apei subterane prin discretizarea atât a interiorului domeniului cât și a frontierei domeniului. Această discretizare conduce la erori care au fost prezentate și comentate cu programele respective.

Așa cum s-a văzut dificultățile apar în vecinătatea singularităților (ex. puțuri) unde relația logaritmică a denivelării nu poate fi modelată prin discretizare cu suficientă acuratețe. Erorile sunt dependente de discretizarea domeniului fără a avea posibilitatea unei evaluări exacte, exceptând cazurile simple.

Metoda elementelor de frontieră dezvoltată în mod deosebit într-o perioadă relativ recentă este o metodă numerică de rezolvare a problemelor la limită. În esență această metodă, utilizând o soluție fundamentală a ecuației cu derivate parțiale date conține implicit comportamentul singular și nu necesită discretizare de domeniu. Problema rezolvării numerice a ecuației integrale la care conduce reprezentarea soluției necesită doar o discretizare a frontierei. Se pot obține astfel datele necesare care vor permite, prin intermediul unei reprezentări integrale numerice asociate ecuației date, calculul soluției în orice punct al domeniului. Soluțiile astfel obținute așa cum s-a mai menționat conțin comportamentul singular real în vecinătatea puțurilor și deci nu sunt afectate de erori semnificative.

Problema analizată a fost și în acest caz compararea rezultatelor modelării numerice cu soluția obținută analitic.

Ca și exemplu numeric pentru studiul erorii s-a luat în considerare un puț având raza $r_0 = 0.1m$, o denivelare de 1m la o distanță de 10m de puț. Discretizarea pentru MEFRO (fig.4.6) a fost relativ grosieră, doar 12 noduri (elemente) pe contur.

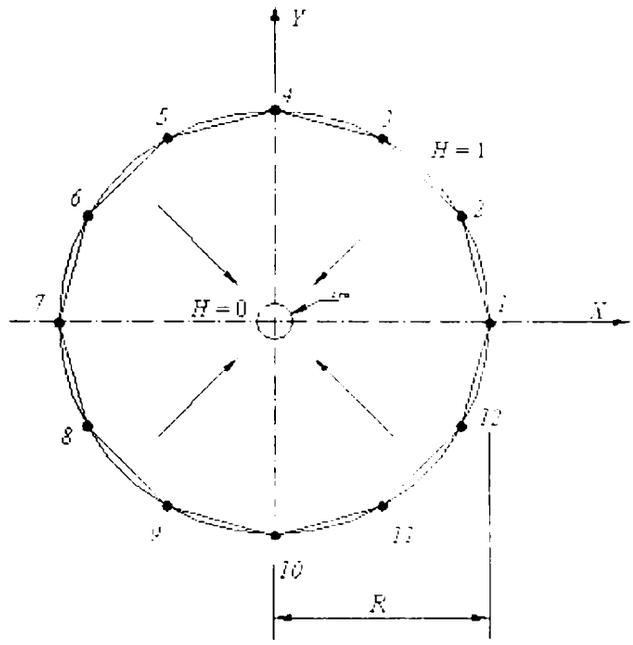


Fig. 4.6 Discretizarea frontierei în cazul unui puț cu denivelare constantă

Valoarea debitului calculat analitic s-a făcut cu ajutorul formulei cunoscute a puțului singular:

$$Q_A = \frac{2\pi kmS}{\ln \frac{R}{r_0}}$$

unde semnificația mărimilor care intervin este următoarea:

- Q_A debitul puțului calculat analitic;
- km transmisivitatea domeniului;
- R raza de influență corespunzător denivelării S;
- r_0 raza puțului;
- S denivelarea.

De asemenea au fost calculate valorile fluxului şi a potențialului în anumite puncte din interiorul domeniului atât numeric cât şi analitic pentru a putea fi comparate în vederea stabilirii erorii MEFRO. Rezultatele sunt prezentate în tabelele 13 şi 14.

Erori la calculul distribuției fluxului Tabelul 13

Nr. nod	Abscisă (m)	Ordonată (m)	H (m)	Flux calculat pe element	Eroare
1	10	0	1,00	-0,214964	0,479%
2	8.7	5	1,00	-0,215500	0,231%
3	5	8.7	1,00	-0,214968	0,477%
4	0	10	1,00	-0,214969	0,477%
5	-5	8.7	1,00	-0,215501	0,231%
6	-8.7	5	1,00	-0,214965	0,478%
7	-10	0	1,00	-0,214966	0,478%
8	-8.7	-5	1,00	-0,215500	0,231%
9	-5	-8.7	1,00	-0,214970	0,476%
10	0	-10	1,00	-0,214968	0,477%
11	5	-8.7	1,00	-0,215500	0,231%
12	8.7	-5	1,00	-0,214967	0,478%

*) Valoarea analitic calculată a fluxului pe un element al conturului exterior este:
q = - 0,216

Erori ale înălțimii piezometrice Tabelul 14

Nr. punct	Abscisa (m)	Ordonata (m)	$H_{Numeric}$ (m)	$H_{Analitic}$ (m)	Eroare
1	1,00	0,00	0,3516	0,3484	0,918%
2	2,00	0,00	0,5030	0,4984	0,922%
3	3,00	0,00	0,6544	0,6484	0,925%
4	4,00	0,00	0,7430	0,7362	0,923%
5	5,00	0,00	0,8059	0,7985	0,926%
6	6,00	0,00	0,8945	0,8862	0,936%
7	8,00	0,00	0,9573	0,9485	0,927%
8	10.0	0.00	1,0000	1,0000	0

4.3.1 Concluzii cu privire la erorile metodei elementelor de frontieră.

În cazul aplicării metodei elementelor de frontieră pentru un puț izolat se obține o precizie de calcul net superioară comparativ cu celelalte metode (MEDIF,MEFIN) prezentate în paragrafele anterioare. În cazul acestei metode nu este nevoie de o discretizare a întregului domeniu ci doar a frontierei sale, ceea ce conduce la formarea unui sistem cu un număr redus de ecuații comparativ cu primele două metode. Pentru exemplul considerat mai sus avem 12 elemente față de sute de elemente în cazul celorlalte metode numerice.

4. 4 Concluzii finale privind analiza erorilor metodelor numerice în vecinătatea captărilor subterane

În cazul **metodei diferențelor finite** ca o concluzie generală se menționează faptul că prin îndesirea rețelei de discretizare este posibilă o reducere a erorilor care însă pot rămâne relativ mari şi pentru rețele dense de discretizare. Astfel în cazul puțurilor cu denivelare dată (înălțimi piezometrice cunoscute) erorile sunt mari chiar şi pentru rețele relativ dense de discretizare. La un număr de 300 de elemente rezultă valori ale erorilor cuprinse între 46% şi 8%. În cazul puțurilor imperfecte erorile introduse de metoda diferențelor finite se situează în limita 30-53%, deci valori foarte mari.

În consecință rafinarea discretizării rețelei prin creşterea numărului de elemente constituente sau introducerea unor funcții de interpolare de grad superior nu conduce la o micşorare mulțumitoare a erorilor. Ea conduce însă la creşterea excesivă a numărului de ecuații şi astfel la mărirea dimensiunilor sistemului de ecuații deci a timpului de calcul.

Metoda elementelor finite introduce în general erori mai mici de calcul decât metoda diferențelor finite. Pentru a obține o precizie de calcul ridicată se constată şi în acest caz că este necesară o discretizare foarte rafinată a rețelei de elemente finite mai ales în vecinătatea singularității.

Desigur problemele pe care le ridică realizarea programelor de calcul în cadrul diferitelor metode numerice au complexitatea lor specifică. Metodele cu diferențe finite şi metoda elementelor finite implică o complexitate scăzută ca şi nivel de programare datorită particularității matricei sistemului (matrice diagonală şi simetrică). În cazul metodei elementelor de frontieră matricea sistemului de ecuații care are o structură compactă, care are totodată ocupării unei foarte mari părți din memoria calculatorului.

Comparând metoda diferențelor finite şi a elementelor finite cu **metoda elementelor de frontieră** sub aspectul implementării lor, se menționează faptul că metoda elementelor de frontieră nu necesită algoritmi speciali pentru realizarea rețelei de discretizare a domeniului. În schimb matricea sistemului final de ecuații în cazul metodei elementelor de frontieră este o matrice compactă cu complicații de implementare. În acelaşi timp, aşa după cum s-a văzut această metodă reprezintă avantaje nete în reprezentarea singularităților. În altă ordine de idei MEDIF şi MEFIN permit modelarea mediilor heterogene deci au un avantaj deosebit a lor în comparație cu MEFRO care este dezvoltat pentru medii omogene sau omogene pe porțiuni. De aceea după cum se va putea observa în capitolele se va insista pentru introducerea diferitelor metode corespunzător specificului lor în modelarea complexă a captărilor subterane.

Ca şi concluzie generală rezultă faptul că metodele numerice trebuie adaptate specificului problemei modelate, recomandându-se chiar combinarea lor.

CAPITOLUL V

MODELAREA MIŞCĂRII ÎN ACVIFERE GENERATE DE SISTEME COMPLEXE DE CAPTARE APLICÂND MEFRO

Dezvoltarea modelării numerice a mişcării apei subterane în ultimul deceniu s-a concentrat îndeosebi asupra modelelor la scară extinsă, regională utilizând metoda diferenţelor finite şi a elementelor finite. (ex. MODFLOW, AQUA, ASM).

Ţinând cont de studiul asupra erorilor celor trei tipuri de metode numerice şi de concluziile finale prezentate în cadrul capitolului IV se poate concluziona faptul că metoda diferenţelor finite şi metoda elementelor finite cu avantaje certe în modelarea la scară regională nu sunt adecvate pentru simularea mişcării în acvifere, generate de sisteme complexe de captare [31].

În general, calculele de hidraulică subterană a mişcării în acvifere presupun folosirea combinată a unor modele care să poată fi aplicate pe domenii întinse, iar altele cum ar fi MEFRO în zone restrânse în vecinătatea captărilor (fig. 5.1).

Astfel pentru domeniul extins (de ordinul I) se recomandă MEDIF sau MEFRO în care un sistem întreg de captare este reprezentat prin debitul global.

Pentru subdomeniul sistemului de captare cu singularităţi ca puţuri, linii de drenaj, subdomenii cu aport de precipitaţii se recomandă calculul folosind metoda elementelor de frontieră, iar pentru puţuri imperfecte se recomandă calculul folosind metoda elementelor analitice [31]

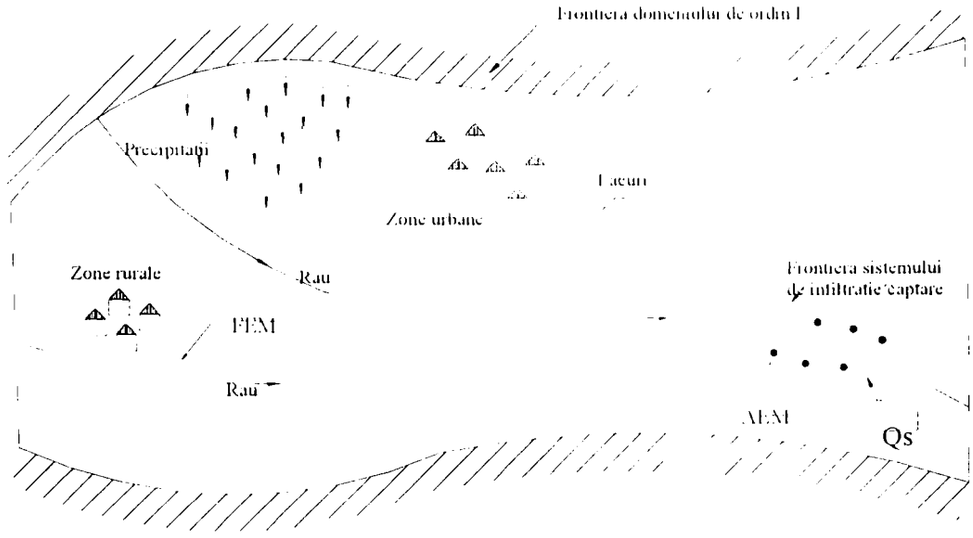


Fig. 5.1 Schiţa unui sistem complex de captare aferentă unui domeniu cu suprafaţă mare [31]

Astfel, pentru frontiera domeniului, singularități, puncte interioare, linii de drenaj, subdomenii cu aport de precipitații din exterior ale unui sistem de captare din cadrul unui sistem hidraulic complex se recomandă calculul folosind metoda elementelor de frontieră, iar pentru puțuri imperfecte se recomandă calculul folosind MEA (fig.5.2).

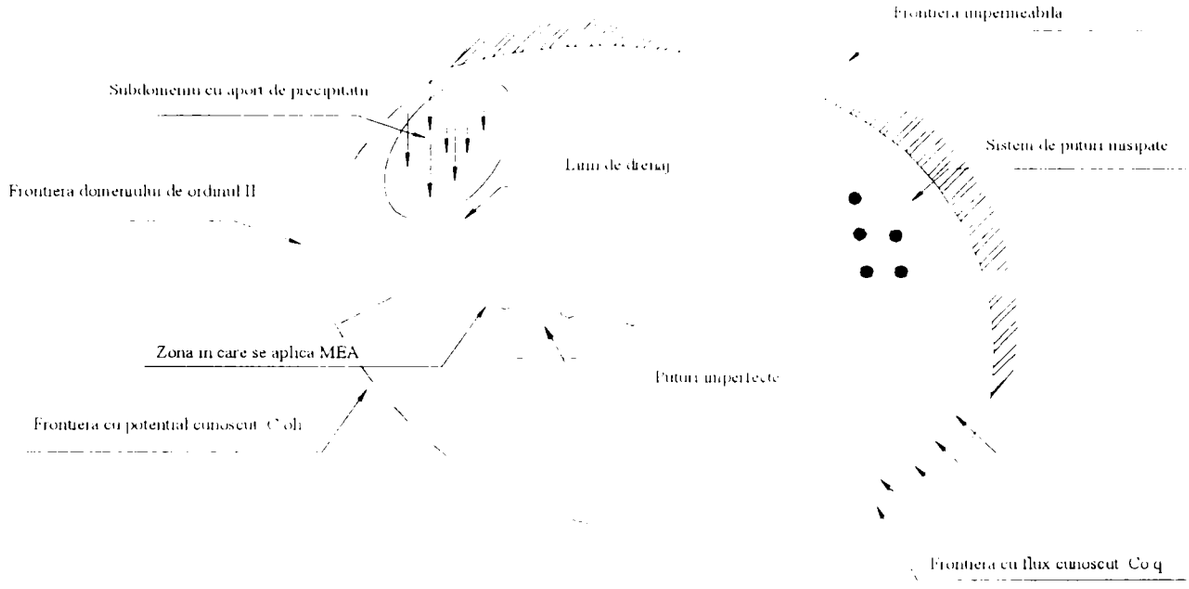


Fig. 5.2 Modul de discretizare al sistemului complex de captare în vecinătatea singularităților

Domeniul de ordinul I, se extinde în plan orizontal având dimensiuni de ordinul kilometrilor sau chiar a zecilor de Km iar domeniul de ordinul II, este de fapt un domeniu semilocal şi care cuprinde unul sau mai multe obiecte de infiltrație/captare.

Prin cuplarea metodelor, respectiv prin folosirea MEFIN pentru domenii întinse de ordinul I, MEFRO şi MEA pentru sistemele complexe de captare din cadrul acestor domenii (de ordinul II) se realizează o cuplare optimală a acestor metode adică o folosire cât mai eficientă a avantajelor fiecăreia [31].

Figura 5.1 prezintă schița unui model la scară largă. În schiță se observă modul în care s-a delimitat sistemul complex de captare care în modelarea de ordinul I se reprezintă cu debitul total al sistemului notat cu Q_s.

Există mai multe posibilități de comportare a acestui sistem (5.1) funcție de debitul Q_s al sistemului [31]. Astfel:

$Q_s < 0 \rightarrow$ avem o alimentare a sistemului din zona extinsă;
$Q_s > 0 \rightarrow$ avem un excedent de debit al sistemului cum ar fi o infiltrare artificială; (5.1)
$Q_s = 0 \rightarrow$ captarea este egală cu alimentarea, adică sistemul este închis cum ar fi cazul unei infiltrări/alimentări în echilibru

În general ce-a de-a treia situație apare destul de rar în realitate.

5.1 Descrierea schemei generale luate în studiu pentru MEFRO.

Modelul matematic folosit pentru studiul sistemelor complexe de captare bazat pe metoda MEFRO/MEA conține următoarele obiecte matematice:

- puțuri cu flux cunoscut (2D);
- puțuri cu potențial cunoscut (2D);
- linii de drenaj (2D);
- puțuri imperfecte (local - 3D);
- puțuri imperfecte (cu luarea în considerare a mişcării interioare cu pierderi de sarcină, 3D);
- subdomenii cu aport de debit (2D).
- frontiera (2D)
- puncte interioare din domeniu unde se cere să se calculeze înălțimea piezometrică (2D);

Condițiile la limită pe frontiera domeniului sunt:

- frontiera cu potențial cunoscut C_{H0F};
- frontieră cu flux cunoscut C_{0q} sau impermeabilă $C_{0\Sigma}$.

Aceste condiții la limită vor fi furnizate de modelarea de ordinul I.

5.2 Reprezentarea integrală indirectă a soluției.

Reprezentarea integrală indirectă a potențialului φ într-un punct M al domeniului D este de forma [31]:

$$\varphi^{+}(M) = -\frac{1}{2\pi}\left[\int_{C_0}\psi(P)G(M,P)dl + \int_{L_{obj}} q_{obj}(P)G(M,P)dl + \sum_{J=1}^{N_W} Q_{wj}G(M,W_j) + \right.$$
$$\left. + \sum_{J=1}^{N_{PW}}\sum_{i=1}^{N_e^{(J)}} q_i^{(J)}G_i^{(J)}(M,W_{p,J})\right] + \int_{D_\varepsilon}\varepsilon(P)G(M,P)d\Omega \qquad (5.1a)$$

unde notațiile folosite sunt următoarele:

C_0 - frontiera domeniului;
L_{obj} - linii de drenaj;
N_{Pw} - numărul de puțuri imperfecte;
$N_e^{(J)}$ - numărul de elemente ale puțului imperfect "j";
N_W - numărul de puțuri cu flux cunoscut;
q_{obj} - debitul specific al liniei de drenaj;
$Q_{W,J}$ - debitul puțului „j";
$q_i^{(J)}$ - debitul specific al unui element al puțului imperfect.

Pentru calculul fluxului:

$$q_{N_M}(M)=\frac{\partial\varphi^{+}(M)}{\partial n_M}=$$

$$=-\frac{1}{2\pi}\left[\int_{C_0^*}\psi(P)F(M,P)dl+\int_{L_{obj}}q_{obj}(P)F(M,P)dl+\sum_{J=1}^{N_W}Q_{wj}F(M,W_j)+\right. \tag{5.1b}$$

$$\left.+\sum_{J=1}^{N_{PW}}\sum_{}^{N_c^{(J)}}q_i F_i^{(J)}(M,W_{Pj})+\int_{D_\varepsilon}\varepsilon(P)F(M,P)d\Omega\right]$$

Primul termen al reprezentărilor integrale (5.1 a,b) corespunde frontierei domeniului. Al doilea termen reprezintă potențialul generat de obiectele orizontale cum sunt de exemplu liniile de drenaj. Cel de-al treilea termen este potențialul generat de puțurile perfecte amplasate în punctele W_j, având debitul Q_{Wj}. Al patrulea termen reprezintă potențialul generat de puțurile imperfecte, iar ultimul corespunde debitului de alimentare distribuit pe suprafața D_ε inclusă în D. Se observă că față de cazul simplu prezentat la bazele metodei în paragraful 2.3, în reprezentările de mai sus s-au introdus s-au introdus obiectele orizontale (linii de drenaj) şi puțurile parțial penetrate. Astfel (5.1a,b) constituie cele mai generale reprezentări integrale care conțin toate obiectele care pot apare în cazul unui sistem de captări subterane şi au fost stabilite de David I. [31].

În continuare se prezintă mai detaliat reprezentarea puțurilor imperfecte care au constituit obiectele principale ale cercetării în cadrul tezei, unele rezultate fiind publicate David I., Eleş G. [28].

5.3 Reprezentarea puțurilor imperfecte în MEFRO.

Baza tehnicii de reprezentare a mişcării generate de puțuri imperfecte o constituie modelarea lui cu elemente analitice care constau în surse liniare distribuite de-a lungul axului puțului [30] (fig. 5.3).

Astfel pornind de la expresia generală a soluției fundamentale tridimensionale se construieşte următoarea reprezentare de bază:

$$\phi_i(M_j)=-\frac{q_i}{4\pi}\int_{z_i^{(K)}}^{z_{i+1}^{(K)}}\frac{dz}{\sqrt{\left(x_i^{(K)}-x_j\right)^2+\left(y_i^{(K)}-y_j\right)^2+\left(z_i^{(K)}-z_j\right)^2}} \tag{5.2}$$

care reprezintă potențialul generat de elementele puțului $M_i^{(k)},M_{i+1}^{(k)}$ în punctul M_j al domeniului D (fig. 5.3)

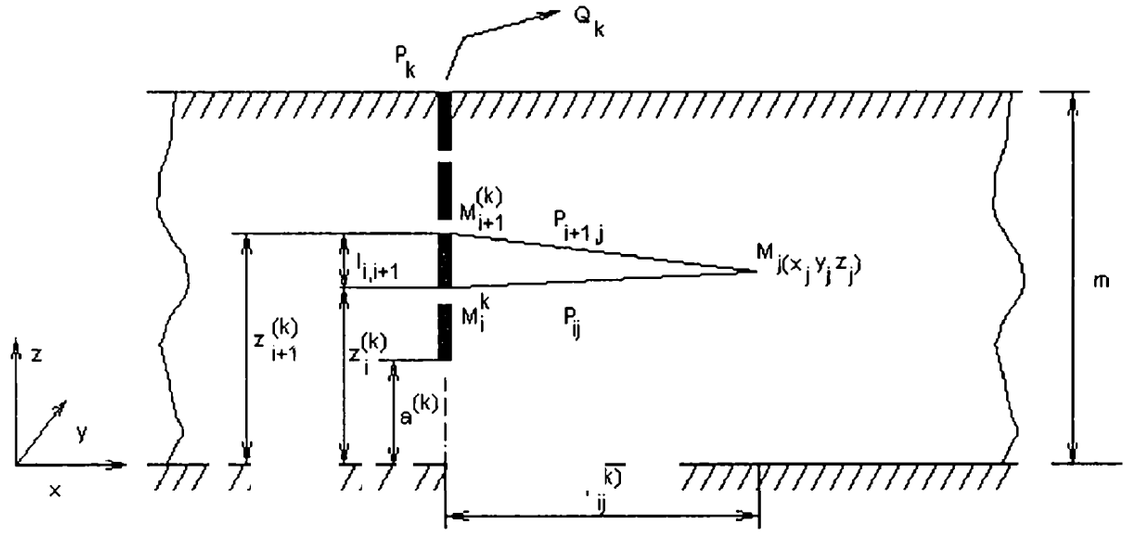


Fig. 5.3 Modul de discretizare al puțurilor imperfecte

Aplicând metoda imaginilor, ceea ce înseamnă oglindirea sursei liniare față de planurile de limitare inferioare şi superioare a acviferului pentru $z = 0$ şi $z = m$ şi calculând suma pentru toate elementele, valoarea potențialului (a înălțimii piezometrice) generată de puțurile imperfecte P_K, $K = 1,2...N_P$ într-un punct al domeniului are următoarea formă:

$$h(M_j) = \sum_{K=1}^{N_P} \sum_{i=1}^{N^{(K)}-1} \frac{q_i}{T} A_{j,i}^{(P_K)} + c \tag{5.3}$$

unde T reprezintă transmisivitatea stratului acvifer $T = k_f m$, $N^{(K)}$ reprezintă numărul de noduri aferente puțului P_K, şi $A_{ji}^{(P_K)}$ reprezintă coeficienții de pondere al elementului $\left(M_i^{(K)}, M_{i+1}^{(K)}\right)$, element constituient al puțului P_K. Formula de calcul pentru $A_{ji}^{(P_K)}$ este prezentată în relația 5.4 unde N_0 reprezintă numărul de oglindiri. Funcțiile F sunt date de relațiile de calcul prezentată în 5.5 şi 5.6.

$$A_{j,i}^{(P_K)} = \frac{m}{4\pi} \left\{ \begin{array}{l} F\left(\rho_{i,j}^{(+)(0)(K)}, \rho_{i+1,j}^{(+)(0)(K)}, l_{i,i+1}^{(K)}\right) + F\left(\rho_{i,j}^{(-)(0)(K)}, \rho_{i+1,j}^{(-)(0)(K)}, l_{i,i+1}^{(K)}\right) + \\ \sum_{n=1}^{N_0} \left[\begin{array}{l} F\left(\rho_{i,j}^{(+)(n)(K)}, \rho_{i+1,j}^{(+)(n)(K)}, l_{i,i+1}^{(K)}\right) + F\left(\rho_{i,j}^{(-)(n)(K)}, \rho_{i+1,j}^{(-)(n)(K)}, l_{i,i+1}^{(K)}\right) + \\ F\left(\rho_{i,j}^{(+)(-n)(K)}, \rho_{i+1,j}^{(+)(-n)(K)}, l_{i,i+1}^{(K)}\right) + F\left(\rho_{i,j}^{(-)(-n)(K)}, \rho_{i+1,j}^{(-)(-n)(K)}, l_{i,i+1}^{(K)}\right) \end{array} \right] - \\ \frac{l_{i,i+1}^{(K)}}{2m} \left[\begin{array}{l} F\left(\rho_{\tilde{i},j}^{(+)(N_0)(K)}, \rho_{\tilde{i},j}^{(-)(-N_0)(K)}, l_{\tilde{i},\tilde{i}}^{(+)(N_0)(K)}\right) + \\ F\left(\rho_{\tilde{i},j}^{(+)(-N)(K)}, \rho_{\tilde{i},j}^{(-)(N_0)(K)}, l_{\tilde{i},\tilde{i}}^{(-)(N_0)(K)}\right) \end{array} \right] + \frac{2l_{i,i+1}^{(K)}}{m} F_{ji\infty} \end{array} \right\} \tag{5.4}$$

unde:

$$F\left(\rho_{i,j}^{(\ )(\ )(\ )},\rho_{i+1,j}^{(\ )(\ )(\ )},l_{i,i+1}^{(\ )}\right)=ln\frac{\rho_{i,j}^{(\ )}+\rho_{i+1,j}^{(\ )}-l_{i,i+1}^{(\ )}}{\rho_{i,j}^{(\ )}+\rho_{i+1,j}^{(\ )}+l_{i,i+1}^{(\ )}} \tag{5.5}$$

$$F_{j|\infty}=ln\frac{r_{ij}^{(K)}}{m} \tag{5.6}$$

$$\rho_{i,j}^{(\pm)(0)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+\left(z_j\ \mathrm{m}z_i^{(K)}\right)^2} \quad ; \quad \rho_{i,j}^{(\pm)(n)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+\left(2nm\pm z_i^{(K)}-z_j\right)^2}$$

$$\rho_{i,j}^{(\pm)(-n)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+\left(-2nm\pm z_i^{(K)}-z_j\right)^2} \quad ; \quad \rho_{i,j}^{(0)(0)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+z_j^2}$$

$$\rho_{\tilde{i},j}^{(\mp)(-N_0)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+\left[-(2N_0+1)m\mp\frac{z_i^{(K)}+z_{i+1}^{(K)}}{2}\right]^2};$$

$$\rho_{\tilde{i},j}^{(\pm)(-N_0)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+\left[(2N_0+1)m\pm\frac{z_i^{(K)}+z_{i+1}^{(K)}}{2}-z_j\right]^2};$$

$$l_{\tilde{i},\tilde{j}}^{(\pm)(N_0)(K)}=(2N_0+1)m\pm\frac{z_i^{(K)}+z_{i+1}^{(K)}}{2};$$

$$r_{i,j}^{(K)}=\sqrt{\left(x_i^{(K)}-x_j\right)^2+\left(y_i^{(K)}-y_j\right)^2}; \tag{5.7}$$

Calculul termenilor $\rho_{\tilde{i},j}^{(\ )(\ )(\ )}$ şi $l_{\tilde{i},\tilde{i}}^{(\ )(\ )(\ )}$ se va face identic ca şi în cazul calculului lui $\rho_{i,j}^{(\ )(\ )(\ )}$ şi $l_{i,i+1}^{(\ )(\ )(\ )}$ cu deosebirea că se vor lua în considerare valorile medii iar $\rho_{i+1,j}^{(\ )(\ )(\ )}$ se va calcula tot cu aceeaşi relaţie înlocuind pe $z_i^{(K)}$ cu $z_{i+1}^{(K)}$.

În cazul când se iau în considerare şi pierderile interioare în puţ, cazul înnisipării se consideră mişcarea din interiorul puţului în condiţiile când acesta este plin cu nisip pierderea de sarcină pentru un element $j, j+1$, al puţului se calculează cu relaţia:

$$\Delta h_j=\left(Q_{put}-\sum_{i=1}^{j-1}q_i l_i+q_j\frac{l_j}{2}\right)\lambda l_j \tag{5.8}$$

unde valoarea coeficientului λ are expresia:

$$\lambda=\frac{k_f m}{k_0 A} \tag{5.9}$$

Semnificația mărimilor din relații fiind următoarea:

m - grosimea stratului acvifer;
k_f - coeficientul de filtrație al stratului acvifer;
k_0 - coeficientul de filtrație al materialului (nisipului) din interiorul puțului;
A - aria secțiunii puțului $A = \pi r_0^2$;
Q - debitul extras din puț;
q_i - debitul infiltrat în puț

Calculul trebuie să se efectueze iterativ pornindu-se de la o valoare dată a potențialului puțului. Introducerea pierderilor se realizează prin intermediul condițiilor la limită şi anume H_0 în puț. Valoarea nouă calculată a potențialului se calculează ținând cont de pierderea de sarcină calculată anterior pentru fiecare element al puțului:

$$H_{nou} = H_{vechi} - \left(\sum_{i=n}^{i=j+1} \Delta h_i + \frac{\Delta h_j}{2} \right) \tag{5.10}$$

Această iterație este necesară deoarece Δh_j este dependentă de distribuția debitului "q_i" de-a lungul puțului 5.8 necunoscută apriori.

5.4 Alcătuirea sistemului de ecuații pentru reprezentările generale

Analizând la bază reprezentările integrale indirecte 5.1a, 5.1b prezentate în cadrul subcapitolului 5.2 prin trecere la limită şi utilizând condițiile la limită (pe frontieră şi pe obiectele sistemului de captare) se obține un sistem de ecuații integrale.

Trecând în continuare la discretizare cu:

N_1 – numărul de noduri pentru conturul general;
N_3 – numărul de noduri pentru liniile de drenaj;
N_4 – numărul de puțuri cu debit cunoscut;
N_5 – numărul de puțuri cu potențial cunoscut;
N_6 – numărul de subdomenii cu precipitații date;
$M_{(v)}$ – numărul de elemente al subdomeniului k;
N_7 – numărul de puțuri imperfecte;
N_8 – numărul total de elemente ale puțurilor imperfecte

Ecuațiile integrale conduc la un sistem de ecuații liniare care are următoarea formă generală:

- Ecuațiile pentru elemente de frontieră $P_j - P_{j+1}$ cu potențial cunoscut $H = H_{0j}$:

$$\sum_{\substack{i=1\\ i\neq j}}^{N_1} a_{ji}\frac{\psi_i}{TS_0} + \frac{\psi_j}{TS_0}a_{jj} + \sum_{i=1}^{N_5}\lambda_{ji}\frac{Q_i}{TS_0} + \sum_{i=1}^{N_3-1} a_{ji}\frac{q_i}{TS_0} + \sum_{k=1}^{N_7}\sum_{i=1}^{N_8-1} s_{ji}\frac{q_i}{TS_0} + c = \frac{H_j^0}{S_0} + \sum_{v=1}^{N_6}\frac{\varepsilon_v}{TS_0}\sum_{k=1}^{M_v} c_{ji} - \sum_{i=1}^{N_4}\lambda_{ji}\frac{Q_i}{TS_0} \tag{5.11}$$

- Ecuațiile pentru elemente de frontieră $P_j - P_{j+1}$ cu flux cunoscut $\Delta Q_{j,j+1}$:

$$\sum_{\substack{i=1\\ i\neq j}}^{N_1} b_{ji}\frac{\psi_i}{TS_0} - \frac{1}{2}\frac{\psi_j}{TS_0} + \sum_{i=1}^{N_5}\mu_{ji}\frac{Q_i}{TS_0} + \sum_{i=1}^{N_3-1} b_{ji}\frac{q_i}{TS_0} + \sum_{k=1}^{N_7}\sum_{i=1}^{N_8-1} t_{ji}\frac{q_i}{TS_0} = \frac{\Delta Q_{j,j+1}}{TS_0 l_{j,j+1}} + \sum_{v=1}^{N_6}\frac{\varepsilon_v}{TS_0}\sum_{i=1}^{M_v} d_{ji} + \sum_{i=1}^{N_4}\mu_{ji}\frac{Q_i}{TS_0} \tag{5.12}$$

- Ecuațiile pentru puțuri cu potențial cunoscut H_{Wj}:

$$\sum_{i=1}^{N_1} a_{ji}\frac{\psi_i}{TS_0} + \sum_{\substack{i=1\\ i\neq j}}^{N_5} f_{ji}\frac{Q_i}{TS_0} + f_{jj}\frac{Q_j}{TS_0} + \sum_{i=1}^{N_3-1} a_{ji}\frac{q_i}{TS_0} + \sum_{k=1}^{N_7}\sum_{i=1}^{N_8-1} s_{ji}\frac{q_i}{TS_0} + c = \frac{H_{W_j}}{S_0} + \sum_{v=1}^{N_6}\frac{\varepsilon_v}{TS_0}\sum_{k=1}^{M_v} c_{jk} - \sum_{i=1}^{N_4} f_{ji}\frac{Q_i}{TS_0} \tag{5.13}$$

- Ecuațiile pentru linii de drenaj cu potențial cunoscut $H = H_l$:

$$\sum_{i=1}^{N_1} a_{ji}\frac{\psi_i}{TS_0} + \sum_{i=1}^{N_5}\lambda_{ji}\frac{Q_i}{TS_0} + \sum_{\substack{i=1\\ i\neq j}}^{N_3-1} a_{ji}\frac{q_i}{TS_0} + a_{jj}\frac{q_i}{TS_0} + \sum_{k=1}^{N_7}\sum_{i=1}^{N_8-1} s_{ji}\frac{q_i}{TS_0} + c = \frac{H_l}{S_0} + \sum_{v=1}^{N_6}\frac{\varepsilon_v}{TS_0}\sum_{k=1}^{M_v} c_{jk} - \sum_{i=1}^{N_4}\lambda_{ji}\frac{Q_i}{TS_0} \tag{5.14}$$

- Ecuațiile pentru puțurile imperfecte cu potențial cunoscut $H = H_j$:

$$\sum_{i=1}^{N_1} a_{ji}\frac{\psi_i}{TS_0} + \sum_{i=1}^{N_5}\lambda_{ji}\frac{Q_i}{TS_0} + \sum_{i=1}^{N_3-1} a_{ji}\frac{q_i}{TS_0} + \sum_{k=1}^{N_7}\sum_{i=1}^{N_8-1} s_{ji}\frac{q_i}{TS_0} + c = \frac{H_j}{S_0} + \sum_{v=1}^{N_6}\frac{\varepsilon_v}{TS_0}\sum_{k=1}^{M_v} c_{jk} - \sum_{i=1}^{N_4}\lambda_{ji}\frac{Q_i}{TS_0} \tag{5.15}$$

- Ecuația de bilanț:

$$\sum_{i=1}^{N_1} l_{i,i+1}\frac{\psi_i}{TS_0} + \sum_{i=1}^{N_5}\frac{Q_i}{TS_0} + \sum_{i=1}^{N_3-1} l_{i,i+1}\frac{q_i}{TS_0} + \sum_{k=1}^{N_7}\sum_{i=1}^{N_8-1} l_{i,i+1}\frac{q_i}{TS_0} + c = \sum_{i=1}^{N_6}\frac{\varepsilon_i A_i}{TS_0} - \sum_{i=1}^{N_4}\frac{Q_i}{TS_0} \tag{5.16}$$

Fig. 5.4 Ecuațiile constituente ale sistemului final de ecuații

Expresia coeficienților care intervin în alcătuirea acestor ecuații este următoarea:

$$a_{ji}=\frac{r_i^{(j)}}{2\pi}\left[\left(ln\frac{r_i^{(j)}}{r_0}-1\right)cos\alpha_i^{(j)}+\left(\frac{\pi}{2}-\alpha_i^{(j)}\right)sin\alpha_i^{(j)}\right]+\frac{r_{i+1}^{(j)}}{2\pi}\left[\left(ln\frac{r_{i+1}^{(j)}}{r_0}-1\right)cos\alpha_{i+1}^{(j)}+\left(\frac{\pi}{2}-\alpha_{i+1}^{(j)}\right)sin\alpha_{i+1}^{(j)}\right] \tag{5.17}$$

$$b_{ji}=\frac{1}{2\pi}\left(\alpha_i^j+\alpha_{i+1}^j-\pi\right)cos\theta_{ji}+\frac{1}{2\pi}sin\theta_{ji}\,ln\left|\frac{sin\alpha_{i+1}^{(j)}}{sin\alpha_i^{(j)}}\right| \tag{5.18}$$

$$\theta_{ji}=\alpha_{j,j+1}-\alpha_{i,i+1} \tag{5.19}$$

$$\mu_{ji}=\frac{1}{2\pi}\frac{\left(x_j-x_i\right)sin\alpha_{j,j+1}-\left(y_j-y_i\right)cos\alpha_{j,j+1}}{\left(x_j-x_i\right)^2+\left(y_j-Y_i\right)^2} \tag{5.20}$$

$$r_i^{(j)}=\sqrt{\left(x_i-\frac{x_j-x_{j+1}}{2}\right)^2+\left(y_i-\frac{y_j-y_{j+1}}{2}\right)^2} \tag{5.21}$$

$$c_{jk}=\frac{\left(r_k^{(j)}\,sin\alpha_k^{(j)}\right)^2}{4\pi}\left[\left(ln\frac{r_k^{(j)}}{r_0}-\frac{3}{2}\right)ctg\alpha_k^{(j)}+\frac{\pi}{2}-\alpha_k^{(j)}\right]+\frac{\left(r_{k+1}^{(j)}\,sin\alpha_{k+1}^{(j)}\right)^2}{4\pi}\left[\left(ln\frac{r_{k+1}^{(j)}}{r_0}-\frac{3}{2}\right)ctg\alpha_{k+1}^{(j)}+\frac{\pi}{2}-\alpha_{k+1}^{(j)}\right] \tag{5.22}$$

$$d_{jk}=\frac{r_k^{(j)}}{2\pi}\left|sin\alpha_k^{(j)}\right|\left[sin\theta_{kj}\,ln\frac{r_k^{(j)}}{r_{k+1}^{(j)}}+\left(\alpha_k^{(j)}+\alpha_{k+1}^{(j)}-\pi\right)cos\theta_{kj}\right] \tag{5.23}$$

$$\alpha_i^j = \begin{cases} \left|\alpha_{ij} - \alpha_{i,i+1}\right| & \text{pentru } \left|\alpha_{ij} - \alpha_{i,i+1}\right| \le \pi \\ 2\pi - \left|\alpha_{ij} - \alpha_{i,i+1}\right| & \text{pentru } \left|\alpha_{ij} - \alpha_{i,i+1}\right| > \pi \end{cases} \tag{5.24}$$

$$\lambda_{ji} = \frac{1}{2\pi} ln \sqrt{\left(x_i - x_{\tilde{j}}\right)^2 + \left(y_i - y_{\tilde{j}}\right)^2} \tag{5.25}$$

$$x_{\tilde{j}} = \frac{x_j + x_{j+1}}{2} \quad ; \quad y_{\tilde{j}} = \frac{y_j + y_{j+1}}{2} \tag{5.26}$$

$$f_{ji} = \frac{1}{2\pi} ln \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2} \quad ; \quad f_{jj} = \frac{1}{2\pi} ln \frac{d_0}{2} \tag{5.27}$$

$$s_{j,i}^{(P_K)} = \frac{T}{4\pi} \left\{ \begin{array}{l} F\left(\rho_{i,j}^{(+)(0)(K)}, \rho_{i+1,j}^{(+)(0)(K)}, l_{i,i+1}^{(K)}\right) + F\left(\rho_{i,j}^{(-)(0)(K)}, \rho_{i+1,j}^{(-)(0)(K)}, l_{i,i+1}^{(K)}\right) + \sum_{n=1}^{N} \left[\begin{array}{l} F\left(\rho_{i,j}^{(+)(n)(K)}, \rho_{i+1,j}^{(+)(n)(K)}, l_{i,i+1}^{(K)}\right) + F\left(\rho_{i,j}^{(-)(n)(K)}, \rho_{i+1,j}^{(-)(n)(K)}, l_{i,i+1}^{(K)}\right) + \\ F\left(\rho_{i,j}^{(+)(-n)(K)}, \rho_{i+1,j}^{(+)(-n)(K)}, l_{i,i+1}^{(K)}\right) + F\left(\rho_{i,j}^{(-)(-n)(K)}, \rho_{i+1,j}^{(-)(-n)(K)}, l_{i,i+1}^{(K)}\right) \end{array} \right] - \\ \frac{l_{i,i+1}^{(K)}}{2T} \left[F\left(\rho_{\tilde{i},j}^{(0)(0)(K)}, \rho_{N,j}^{(+)(N)(K)}, l_{\tilde{i},N}^{(+)(K)}\right) + F\left(\rho_{\tilde{i},j}^{(0)(0)(K)}, \rho_{N,j}^{(-)(N)(K)}, l_{\tilde{i},N}^{(-)(K)}\right) \right] + \frac{2l_{i,i+1}^{(K)}}{T} F_{ji\infty} \end{array} \right\}$$

$$l_{j,i}^{(P_K)}=\frac{T}{4\pi}\left\{\begin{array}{l}G\left(\rho_{i,j}^{(+)(0)(K)},\rho_{i+1,j}^{(+)(0)(K)},l_{i,i+1}^{(K)}\right)+G\left(\rho_{i,j}^{(-)(0)(K)},\rho_{i+1,j}^{(-)(0)(K)},l_{i,i+1}^{(K)}\right)+\sum_{n=1}^{N}\left[\begin{array}{l}G\left(\rho_{i,j}^{(+)(n)(K)},\rho_{i+1,j}^{(+)(n)(K)},l_{i,i+1}^{(K)}\right)+G\left(\rho_{i,j}^{(-)(n)(K)},\rho_{i+1,j}^{(-)(n)(K)},l_{i,i+1}^{(K)}\right)+\\ G\left(\rho_{i,j}^{(+)(-n)(K)},\rho_{i+1,j}^{(+)(-n)(K)},l_{i,i+1}^{(K)}\right)+G\left(\rho_{i,j}^{(-)(-n)(K)},\rho_{i+1,j}^{(-)(-n)(K)},l_{i,i+1}^{(K)}\right)\end{array}\right]-\\ \frac{l_{i,i+1}^{(K)}}{2T}\left[G\left(\rho_{\tilde{i},j}^{(0)(0)(K)},\rho_{N,j}^{(+)(N)(K)},l_{\tilde{i},N}^{(+)(K)}\right)+G\left(\rho_{\tilde{i},j}^{(0)(0)(K)},\rho_{N,j}^{(-)(N)(K)},l_{\tilde{i},N}^{(-)(K)}\right)\right]+\frac{2l_{i,i+1}^{(K)}}{T}G_{ji\infty}\end{array}\right\}$$

$$F\left(\rho_{i,j}^{(\ )(\ )(\ )},\rho_{i+1,j}^{(\ )(\ )(\ )},l_{i,i+1}^{(\ )}\right)=ln\frac{\rho_{i,j}^{(\ )}+\rho_{i+1,j}^{(\ )}-l_{i,i+1}^{(\ )}}{\rho_{i,j}^{(\ )}+\rho_{i+1,j}^{(\ )}+l_{i,i+1}^{(\ )}} \tag{5.30}$$

$$F_{ji\infty}=ln\frac{r_{ij}^{(K)}}{m} \tag{5.31}$$

$$\rho_{i,j}^{(\pm)(0)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+\left(z_j\ \mathrm{m}z_i^{(K)}\right)^2}\quad;\quad \rho_{i,j}^{(\pm)(n)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+\left(2nm\pm z_i^{(K)}-z_j\right)^2}\ ; \tag{5.32}$$

$$\rho_{i,j}^{(\pm)(-n)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+\left(-2nm\pm z_i^{(K)}-z_j\right)^2}\quad;\quad \rho_{i,j}^{(0)(0)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+z_j^2}\ ;$$

$$\rho_{\tilde{i},j}^{(\pm)(-N_0)(K)}=\sqrt{\left(r_{i,j}^{(K)}\right)^2+\left[(2N_0+1)m\pm\frac{z_i^{(K)}+z_{i+1}^{(K)}}{2}-z_j\right]^2}\ ;$$

$$l_{\tilde{i},\tilde{i}}^{(\pm)(N_0)(K)}=(2N_0+1)m\pm\frac{z_i^{(K)}+z_{i+1}^{(K)}}{2}\ ;$$

$$r_{i,j}^{(K)}=\sqrt{\left(x_i^{(K)}-x_j\right)^2+\left(y_i^{(K)}-y_j\right)^2}\ ;$$

Calculul termenilor $\rho_{\tilde{i},j}^{(\ )(\ )(\ )}$ şi $I_{\tilde{i},\tilde{l}}^{(\ )(\ )(\ )}$ se va face identic ca şi în cazul calculului lui $\rho_{i,j}^{(\ )(\ )(\ )}$ şi $I_{i,l+1}^{(\ )(\ )(\ )}$ cu deosebirea că se vor lua în considerare valorile medii iar $\rho_{i+1,j}^{(\ )(\ )(\ )}$ se va calcula tot cu aceeaşi relaţie înlocuind pe $z_i^{(K)}$ cu $z_{i+1}^{(K)}$.

În final numărul de ecuaţii al sistemului este:

$$dim = N_1 + \left(N_3^k - 1\right) + N_4 + N_5 + \sum_{k=1}^{N_7}\left(N_8^k - 1\right) + \sum_{k=1}^{N}\left(M_\nu - 1\right) + 1 \tag{5.33}$$

Forma finală matriceală a sistemului de ecuaţii este:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & & a_{1dim} \\ a_{21} & a_{22} & a_{23} & & a_{2dim} \\ & & \vdots & & \vdots \\ a_{dim1} & a_{dim2} & a_{dim3} & \dots & a_{1dim} \end{pmatrix} \times \begin{pmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_{dim} \end{pmatrix} = \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_{dim} \end{pmatrix} \tag{5.34}$$

Prin rezolvarea numerică a sistemului de ecuaţii se obţin următoarele necunoscute:

- valorile funcţiei de distribuţie ψ_j pentru fiecare element de frontieră;
- valorile Q_k pentru puţurile cu potenţial cunoscut;
- valorile q_j corespunzătoare elementelor liniilor de drenaj;
- valorile q_j corespunzătoare elementelor puţurilor imperfecte;
- valoarea constantei c.

Pentru calculul potenţialului $h_{(P)}$ într-un punct $P\left(x_P, y_P, z_P\right)$ din interiorul domeniului se va folosi relaţia:

$$h(P) = \sum_{j=1}^{N_1} a_{jP}\varphi_j + \sum_{j=1}^{N_4} d_{jP} + \sum_{j=1}^{N_5} c_{jP}\varphi_j + \sum_{k=1}^{N_2}\sum_{j=1}^{N_3-1} a_{jp}\varphi_j + \sum_{k=1}^{N_7}\sum_{j=1}^{N_8-1} s_{jP}\varphi_j + \varphi_j \tag{5.35}$$

unde:

$$c_{jP} = \frac{1}{2\pi} ln\sqrt{\left(x_j - x_P\right)^2 + \left(y_j - y_P\right)^2} \tag{5.36}$$

Aceste baze teoretice şi reprezentări [31] au stat la baza programului **MEFRO** elaborat în cadrul tezei, constituind de altfel contribuţia de bază din teză.

CAPITOLUL VI.

PRINCIPII DE BAZĂ FOLOSITE LA ELABORAREA PROGRAMULUI MEFRO

6.1 Limbaje de programare. Generalități.

Limbajele de programare sunt mijloace de comunicare între utilizator şi sistemul de calcul. Prin intermediul lor programatorul transmite calculatorului ordinele necesare pentru executarea anumitor operații. De aceea, el se numeşte limbaj de programare. Descrierea cu ajutorul unui limbaj de programare a etapelor necesare rezolvării unei anumite probleme se numeşte program.

Limbajul de programare este un limbaj artificial, care ține cont de limitările calculatorului, şi de interesele programatorului, adică de operațiile pe care calculatorul le poate realiza, pe de o parte, şi de raționamentele pe care programatorul le face pentru rezolvarea problemelor sale, pe de altă parte.

Evoluția limbajelor de programare [16], [48] este caracterizată de modificarea gradului de satisfacere a celor două categorii de cerințe, fiind evidentă tendința de apropiere a exprimării programelor de limbajul natural şi de operațiile specifice raționamentului uman.

Este cunoscut faptul că un calculator nu "cunoaşte" decât un singur limbaj – limbajul calculatorului, cod maşină. Primele generații de calculatoare erau programate direct în cod-maşină, lucru care era deosebit de dificil de realizat.

Primul pas în evoluția limbajelor de programare l-a reprezentat limbajul de asamblare care are o legătură directă cu limbajul calculatorului, fiecare instrucțiune fiind corespondenta unei operații elementare a calculatorului.

Corespondența directă între instrucțiunile limbajului de programare şi codul maşină a fost abandonată prin introducerea limbajelor de nivel înalt.

Pentru aceasta, programul compilator traduce fiecare instrucțiune a limbajului de nivel înalt într-un grup de instrucțiuni maşină care au acelaşi efect. Programele scrise în limbajul simbolic de nivel înalt se numesc programe sursă, iar limbajul lor, limbaj sursă. Programele rezultate din traducere se numesc programe obiect, iar limbajul lor, limbaj obiect.

Uneori limbajul obiect nu coincide cu limbajul calculatorului, fiind o formă intermediară între acesta şi limbajul sursă. În astfel de cazuri, între compilare şi execuție se interpune o fază suplimentară, numită editare de legături.

În evoluția limbajelor de programare există câteva repere care trebuie menționate: anul 1955 marchează apariția limbajului FORTRAN, destinat calculelor tehnico-ştiințifice cu caracter numeric.

Un alt punct de referință în evoluția limbajelor de programare simbolice este anul 1960 care marchează apariția limbajului ALGOL-60, a cărui definiție se remarcă prin precizie şi o sintaxă complect formalizată.

Între anii 1960-1970 au apărut multe limbaje de programare de nivel superior. Unele din acestea fiind specializate pe anumite clase de probleme, altele universale. Unele dintre aceste programe pot descrie orice fel de prelucrări. Totuşi, se constată faptul că unele prelucrări se descriu mult mai uşor într-un limbaj decât în altul, putându-se distinge limbaje orientate spre prelucrări numerice, spre prelucrări nenumerice, spre prelucrări în timp real etc.

Anul 1970 a fost marcat de către crearea limbajului C de către Dennis M. Ritchie şi Brian W. Kerningham de la Bell Laboratories, New Jersy, fiind inițial destinat scrierii unei părți a sistemului de operare UNIX.

Limbajul C este un limbaj de nivel înalt, orientat către aplicații aşa cum sunt aproape toate limbajele independente de calculator: FORTRAN, PASCAL etc.

Limbajul C împreună cu funcțiile sale de bibliotecă oferă programatorilor acces la maşină şi la facilitățile sistemului de operare mai mult decât oricare limbaj de programare independent de maşină, prin aceste caracteristici fiind un limbaj pentru profesioniști care au drept profesie realizarea de produse program.

Față de limbajul PASCAL, limbajul C simulează mai mult programarea modulară, prin utilizarea de funcții mici, repartizate în unul sau mai multe fişiere sursă, precum şi prin folosirea unor biblioteci de module sursă sau obiect.

Apariția limbajului PASCAL definit în 1971 de Niklaus Wirth [60] este un rezultat al conceptelor dezvoltate ca urmare a crizei care a caracterizat programarea la sfârşitul anilor 60, când problemele abordate au devenit din ce în ce mai complexe. Astfel, programele construite pentru rezolvarea acestor probleme au devenit atât de complexe încât au început săridice o serie de probleme chiar şi pentru creatorii lor. S-a impus astfel din ce în ce mai clar o "disciplină a programării" care a condus la metoda programării structurate.

6.2 Programarea structurată. Metoda programării structurate. Structuri de date

Un program structurat este format din unități funcționale bine conturate, ierarhizate conform problemei. În interiorul unei astfel de unități, structurarea se manifestă atât la nivelul instrucțiunilor cât şi la nivelul datelor.

Programarea structurată este o metodă independentă de limbajul de programare, ea acționând la nivelul stilului de lucru. Metoda programării structurate [60] implică în primul rând principiul de programare "de sus în jos" – adică descompunerea programului în elemente logice independente, numite module, fiecare modul având o funcție bine definită în cadrul programului.

Un alt principiu al programării structurate derivă din teorema lui Böhm şi Jacopini care arată că orice organigramă se poate construi folosind doar trei tipuri de structuri de control: secvențială (secvența), alternativă (decizia), repetitivă (inelul).

Implementarea cea mai spectaculoasă aparține firmei Borland, [19] care ulterior prin compilatoarele Turbo Pascal, şi ulterior Borland Pascal a revoluționat pur şi simplu domeniul, atât prin performanțele intrinseci ale compilatorului, cât şi prin uşurința de verificare a acestuia.

6.2.1 Structuri de date tip listă liniară.

Lista liniară este o structură dinamică în care toate elementele sunt de acelaşi tip, se află concomitent în memorie şi între ele există o relație de ordine[17]. Uneori, după natura programului, sau în conformitate cu cerințele impuse, numărul elementelor poate varia, ceea ce însă nu este cazul în cadrul programului MEFRO.

Lista se poate defini ca fiind o secvență de mai multe elemente de un anumit tip, numit tip de bază.

Proprietatea listei o reprezintă faptul că elementele pot fi ordonate liniar în funcție de poziția lor în cadrul listei astfel:

- a_i precede pe a_{i+1}, $i = 1,2,...n-1$;
- a_i succede pe a_{i-1}, $i = 2,3,...n$;
- elementul i se află pe a i-a poziție.

Lista liniară în principiu este o structură dinamică de date, ea putând fi implementată dinamic numai în limbajele de programare care posedă mecanismul de alocare dinamică a memoriei.

6.2.2 Tipuri de referință (pointer). Alocarea dinamică a memoriei.

În limbajul Pascal pot fi definite structuri complexe pornind de la tipurile simple. Un astfel de caz este de exemplu cazul tabloului. Tabloul reprezintă o mulțime finită şi ordonată de elemente de acelaşi tip, care are asociat un nume, fiecare element al mulțimii fiind identificat prin intermediul numelui tabloului şi al unuia sau mai multor indici de tip ordinal.
De exemplu tabloul pentru un puț:

```
Type
     matrice_put = Array[1..30,1..30] of Real;
```

Problema esențială este însă faptul că acest tip limitează numărul maxim de elemente permise deoarece tabloul este format dintr-un număr finit de elemente făcând parte în acest fel din cadrul structurilor de date statice cărora li se alocă spațiu pe parcursul compilării.

Pentru evitarea numărului de elemente din cadrul unei mulțimi limbajul Borland Pascal permite utilizarea structurilor de date dinamice. Într-o astfel de structură ocuparea spațiului de memorie nu este cunoscută în momentul compilării programului, deoarece, pe măsură ce acesta se execută, spațiul ocupat de memorie poate să crească sau să scadă.

Structurile dinamice neavând nume, sunt accesibile prin intermediul altor variabile, care conțin adresele lor, şi în felul acesta "le indică". Aceste variabile se numesc pointeri şi formează tipul de referință (tipul pointer). Variabilele de tip pointer sunt variabile statice obişnuite şi au ca valori adrese, care pot fi adresele de început ale unor structuri dinamice sau a unei componente a unei asemenea structuri. Prin urmare accesul la o variabilă dinamică, indicată de o variabilă pointer se realizează prin adresarea indirectă a variabilei pointer.

Variabila dinamică indicată de o variabilă pointer, se va referi, nu printr-un nume propriu, ci prin intermediul pointerului său.

Exemplu: **P** – variabilă pointer;
P^ - variabilă dinamică indicată de pointerul P.

În cadrul programului MEFRO, definirea conturul domeniului şi a celorlalte elemente s-a făcut utilizând structurile dinamice de date.

```
PContur = ^TContur;
TContur = Object
  TipNod    : string[8];
  Abscisa   : string[8];
  Ordonata  : string[8];
  Potential : string[8];
  urm,ant   : PContur;
end;
```

unde ***PContur*** reprezintă tipul pointerului.

Alocarea memoriei pentru o structură dinamică se realizează cu ajutorul instrucțiunii New care are un singur parametru de tip pointer.

6.2.3 Structuri de date folosite pentru conturul domeniului în cadrul programului MEFRO.

Utilitatea tipului pointer devine evidentă în cazul structurilor de date dinamice înlănțuite, structuri care în general se definesc recursiv.

Înlănțuirea se pune în evidență cu ajutorul unor ***informații de legătură*** care la rândul lor pot fi memorate în unul sau mai multe câmpuri ale articolului care conține informațiile ataşate unui element al listei. Evident, ***informațiile propriu-zise*** ataşate nodurilor pot fi memorate de asemenea în unul sau mai multe câmpuri ale unei înregistrări sau ale unui obiect.

Informații propriu-zise	Informații de legătură

```
PContur = ^TContur;
TContur = Object
  TipNod    : string[8];
  Abscisa   : string[8];
  Ordonata  : string[8];
  Potential : string[8];
  urm,ant   : PContur;
end;
```

Orice element constituient al listei care defineşte de fapt conturul domeniului conține astfel informațiile utile sau informațiile propriu-zise şi informațiile de legătură, doi pointeri de înlănțuire la elementul anterior şi la elementul următor.

6.2.4 Caracterul de listă dublu înlănțuită

Lista astfel creată are particularitatea de a fi listă dublu înlănțuită , întrucât în meniul de editare este nevoie ca ea să fie parcursă în ambele sensuri, atât înainte cât şi înapoi (adică atât de la primul element spre ultimul cât şi invers). În acest sens informația de legătură pe care o are un element al listei conține un pointer care indică următorul element din listă dar şi un al doilea pointer care să indice elementul anterior (elementul care se află înaintea elementului considerat curent).

Inserarea la începutul listei, se face folosind procedura ***Conturprim***:

```
Procedure Conturprim;
Begin
     Pct:=nil;
     New(Qct);
     Pct:=Qct;
     Pct^.ant:=nil;
     Qct^.urm:=nil;
     Str(Tip,          Qct^.TipNod);
     Str(x:8:3,        Qct^.Abscisa);
     Str(y:8:3,        Qct^.Ordonata);
     Str(potential:8:3,Qct^.Potential);
end;
```

În cadrul acestei proceduri s-au folosit următoarele notații:

Pct – variabilă tip pointer care indică primul element al listei;
Qct – variabilă de lucru de tip pointer.

Secvența de inserarea la începutul listei este prezentată în cadrul procedurii efectul instrucțiunilor fiind următorul:

- prima instrucțiune realizează alocarea dinamică a spațiului necesar câmpurilor primului element şi inițializarea pointerului Qct cu adresa acestui primului element element al listei;
- a doua instrucțiune modifică conținutul pointerului Pct astfel încât el să indice nodul inserat;
- a treia şi a patra instrucțiune realizează legătura dintre elementul inserat în listă şi cel anterior respectiv cel următor
- următoarele instrucțiuni realizează inițializarea câmpurilor cu informații referitoare la primul element al frontierei.

Inserarea restului de elemente ale listei astfel începute, se face prin intermediul procedurii Conturrest.

```
Procedure Conturrest;
Begin
      New(Rct);
      Rct^.urm:=nil;
      Rct^.ant:=Qct;
      Qct^.urm:=Rct;
      Str(Tip,           Rct^.TipNod);
      Str(x:8:3,         Rct^.Abscisa);
      Str(y:8:3,         Rct^.Ordonata);
      Str(potential:8:3,Rct^.Potential);
      Qct:=Rct;
end;
```

Apelând această procedură se inserează în cadrul listei toate celelalte elemente ale listei elementelor de frontieră.

6.2.5 Programarea orientată spre obiecte.

Una din tendințele naturale în evoluția limbajelor de programare este de a pune în corespondență obiecte ale problemei studiate cu reprezentări cât mai fidele la nivelul limbajului. În acest sens un obiect cum este frontiera unui domeniu care are în compunerea sa noduri de diferite tipuri cu potențial sau flux cunoscut funcție de condițiile de margine poat fi reprezentate printr-o înregistrare de câmpuri de diferite tipuri sau printr-un tablou.

Modul de reprezentare menționat mai sus, înregistrarea şi tabloul sunt, dintr-un anumit punct de vedere însă incomplecte. Ele nu pot însă evidenția anumite valori funcție de diverşi parametrii.

Limbajele orientate spre obiecte elimină aceste neajunsuri. Primul limbaj orientat spre obiecte , Simula, a fost dezvoltat la mijlocul anilor 60. Atât el cât şi următorul sistem reprezentativ pentru programarea orientată spre obiecte, Smalltalk-80 nu au cunoscut o răspândire semnificativă până în anii 80. Parțial aceasta s-a datorat şi absenței unor procesoare puternice care să satisfacă cerințele de performanță. Odată cu ameliorarea suportului tehnologic, limbajele orientate spre obiecte au fost acceptate de tot mai mulți programatori, apărând totodată şi alte limbaje orientate spre obiecte ca C++, Turbo Pascal 6.0 sau Borland Pascal 7.0.

Programarea orientată spre obiecte reprezintă o tehnică de programare, iar un limbaj de programare orientat spre obiecte are mecanismele suport necesare acestui stil de programare. În acest mod caz limbajul este înzestrat cu elemente care face programarea orientată spre obiecte suficient de comodă. Un limbaj nu

suportă un stil de programare dacă scrierea programelor în acest stil este greoaie sau cere programatorului să utilizeze tertipuri sau artificii de programare deosebite. Spre exemplu primele variante ale limbajului Fortran nu suportă programarea structurată, iar limbajul Pascal, aşa cum a fost standardizat iniţial, nu este orientat spre obiecte. Abia odată cu versiunea Turbo Pascal 5.5 au fost introduse în limbaj mecanismele suport ale programării orientate spre obiecte.

6.2.6 Definirea obiectelor în cadrul programului MEFRO.

Cu toate că programarea orientată spre obiecte este un stil de programare fundamental deosebit de cel tradiţional, construcţiile sintactice ale limbajului Borland Pascal care o facilitează, pot fi înţelese ca şi extinderi ale unora mai vechi.

Obiectul conţine declaraţii atât pentru date, cât şi pentru proceduri şi funcţii, într-o formă similară definirii unei înregistrări. Procedurile şi funcţiile declarate într-un obiect se numesc ***metode***. Obiectul cuprinde doar antetul metodelor, blocurile asociate lor urmând a fi specificate separat. ***Câmpurile de date*** sunt definite ca înregistrări, prin selectorul şi tipul fiecăruia. În cele ce urmează se prezintă descrierea obiectului Frontiera din cadrul programului **MEFRO**.

```
Type

     Refpi =^Punct_interior;
     Reffr =^Frontiera;
     RefDr =^DrenLin;
     RefQ  =^SingFlux;
     RefH  =^SingPot;


Frontiera = Object
    Constructor Init;
    Procedure Coordonate(Var int:Integer; Var outX,outY:Real);
    Function Potentialul(Var int:Integer):Real;
    Function TipNod     (Var int:Integer):Integer;
    Destructor Sterge;
     Private
      cap,crt,temp,urm:Reffr;
      Tipul,Nr:Integer;
      X,Y,Potential:Real;
  end;


DrenLin = Object
  Constructor Init;
  Procedure Coordonate (Var int:Integer; Var outXa,outYa,outXb,
                                          outYb:Real);
```

```
  Procedure Parametrii (Var int:Integer; Var outP4,outP5,
                                                outP6:Real);
  Procedure Potentialul(Var int:Integer; Var outZ:Real);
  Function  TipNod     (Var int:Integer):Integer;
  Destructor Sterge;
   Private
    cap,crt,temp,urm:RefDr;
    Tipul,Nr:Integer;
    X,Y,Hh,Par4,Par5,Par6:Real;
end;



Punct_interior = Object
  Constructor Init;
  Procedure Coordonate(Var int:integer;Var outX,outY,outZ:real);
  Destructor Sterge;
  Private
     cap,crt,temp,urm:RefPi;
     Nr:Integer;
     X,Y,Z:Real;
 end;



SingFlux = Object
  Constructor Init;
  Procedure Coordonate(Var int:Integer; Var outX,outY:Real);
  Function Debit(Var int:Integer):Real;
  Function Diam(Var int:Integer):Real;
  Destructor Sterge;
   Private
    cap,crt,temp,urm:RefQ;
    Nr:Integer;
    X,Y,Flux,Diametru:Real;
end;



SingPot = Object
   Constructor Init;
   Procedure Coordonate(Var int:Integer; Var outX,outY:Real);
   Function Pot(Var int:Integer):Real;
   Function Diam(var int:Integer):Real;
   Destructor Sterge;
 Private
   cap,crt,temp,urm:RefH;
   Nr:Integer;
   X,Y,H,Diametru:Real;
 end;
```

Unul din scopurile programării orientate spre obiecte este utilizarea obiectelor ca entități complecte, de sine stătătoare: se recomandă ca nici unul din câmpurile unui obiect să fie direct accesibil utilizatorului, orice operație asupra sa realizându-se numai prin imtermediul metodelor. Metodele trebuie să alcătuiască

un set cât mai complet de operații relative la obiect. O astfel de abordare se poate observa în exemplul obiectului Frontiera prezentat mai sus.

Metodele astfel definite permit testarea şi actualizarea oricărui câmp al obiectului. Este important de remarcat faptul că prin acest mod de definire, metodele elimină necesitatea adresării directe a câmpurilor obiectului. Pentru a asigura o protecție a câmpurilor, începând cu versiunea Borland Pascal 7.0 [20], un avantaj substanțial a fost realizat prin faptul că se permite divizarea declarației obiectului în două secțiuni: una ***publică***, accesibilă din afara obiectului şi una ***privată***, accesibilă doar în cadrul obiectului, inclusiv în declarațiile metodelor sale.

Această facilitate a fost folosită în cadrul programului MEFRO în zona de calcul, la definirea obiectelor Frontiera, Drenlin, Punct_interior, SingFlux, SingPot etc. S-a uzat de această facilitate şi pentru a elimina o eventuală manevrare greşită a câmpurilor şi metodelor prin acțiuni din afara obiectului. Secțiunea privată este specificată prin cuvântul cheie ***Private*** şi se află după secțiunea publică.

CAPITOLUL VII.

ELABORAREA PROGRAMULUI MEFRO PENTRU REZOLVAREA SISTEMELOR DE CAPTĂRI SUBTERANE

7.1 Realizarea meniului de introducere şi editare a datelor folosit în cadrul programului MEFRO.

7.1.1 Meniul Editare date program.

Meniul **Editare date program** (fig.7.2) este un meniu principal şi este folosit la pentru următoarele operații asupra datelor de rulare a programului:

- introducerea de date noi;
- salvarea unui set de date;
- încărcarea unui set de date;
- editarea datelor generale ale unei probleme;
- editarea conturului unui domeniu;
- editarea puțurilor cu potențial cunoscut;
- editarea puțurilor cu flux cunoscut;
- editarea liniilor de drenaj;
- editarea punctelor interioare;
- editarea puțurilor imperfecte:
    - o – parametrii puțurilor
    - o – parametrii elementelor
- editarea subdomeniilor:
    - o – parametrii subdomeniilor
    - o – coordonatele elementelor frontierei subdomeniilor

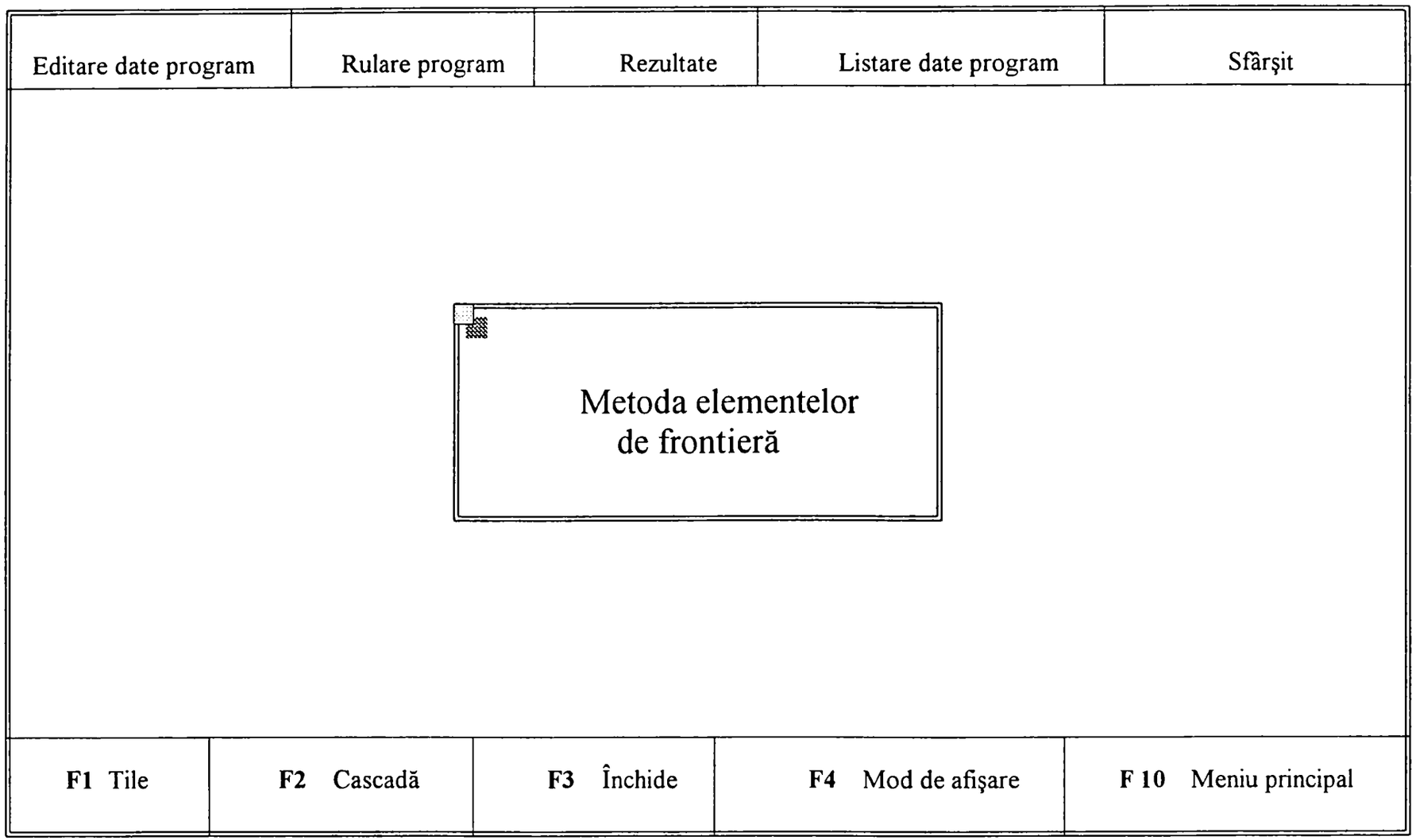


Fig.7.1 Meniul principal al programului MEFRO

7.1.2 Submeniul; Introducere date noi, editarea conturului

Acest submeniu este folosit pentru editarea (introducerea, vizualizarea şi/sau modificarea) datelor care alcătuiesc frontiera domeniului. Utilizatorului i se crează în acest sens facilitatea de a edita următoarele date:

- Tipul nodului;
- Abscisă nod;
- Ordonată nod;
- Potențial nod.

Procedurile şi funcțiile folosite pentru editarea din cadrul acestui submeniu se afla în unitul ***Contur.pas***. În cadrul acestui unit conturul domeniului este definit ca fiind obiectul ***TContur*** în care Tipnod, Abscisa, Ordonata şi Potențialul sunt câmpurile sale.

Totodată obiectul ***TContur*** mai conține în plus şi două elemente ***ant*** şi ***urm*** care sunt de tip referință la obiectul ***TContur***. Deoarece nodurile care alcătuiesc conturul domeniului se află într-o structură de tip listă, elemente de tip referință ***ant*** şi ***urm*** sunt folosite pentru deplasarea şi poziționarea pe un anumit nod din cadrul acestei liste.

Pentru a putea vizualiza câmpurile obiectului ***TContur*** într-o manieră cât mai comodă şi eficientă din punct de vedere al editării, tot în cadrul unitului Contur este definit obiectul ***TBox***.

```
PBox = ^TBox;
  TBox = object(TDialog)
  TipNod,Abscisa,Ordonata,Potential: PInputLine;
  Counter:PcountView;
  constructor Init;
  Procedure HandleEvent(var Event: TEvent);virtual;
  Procedure Retine;
  Procedure Show;
end;
```

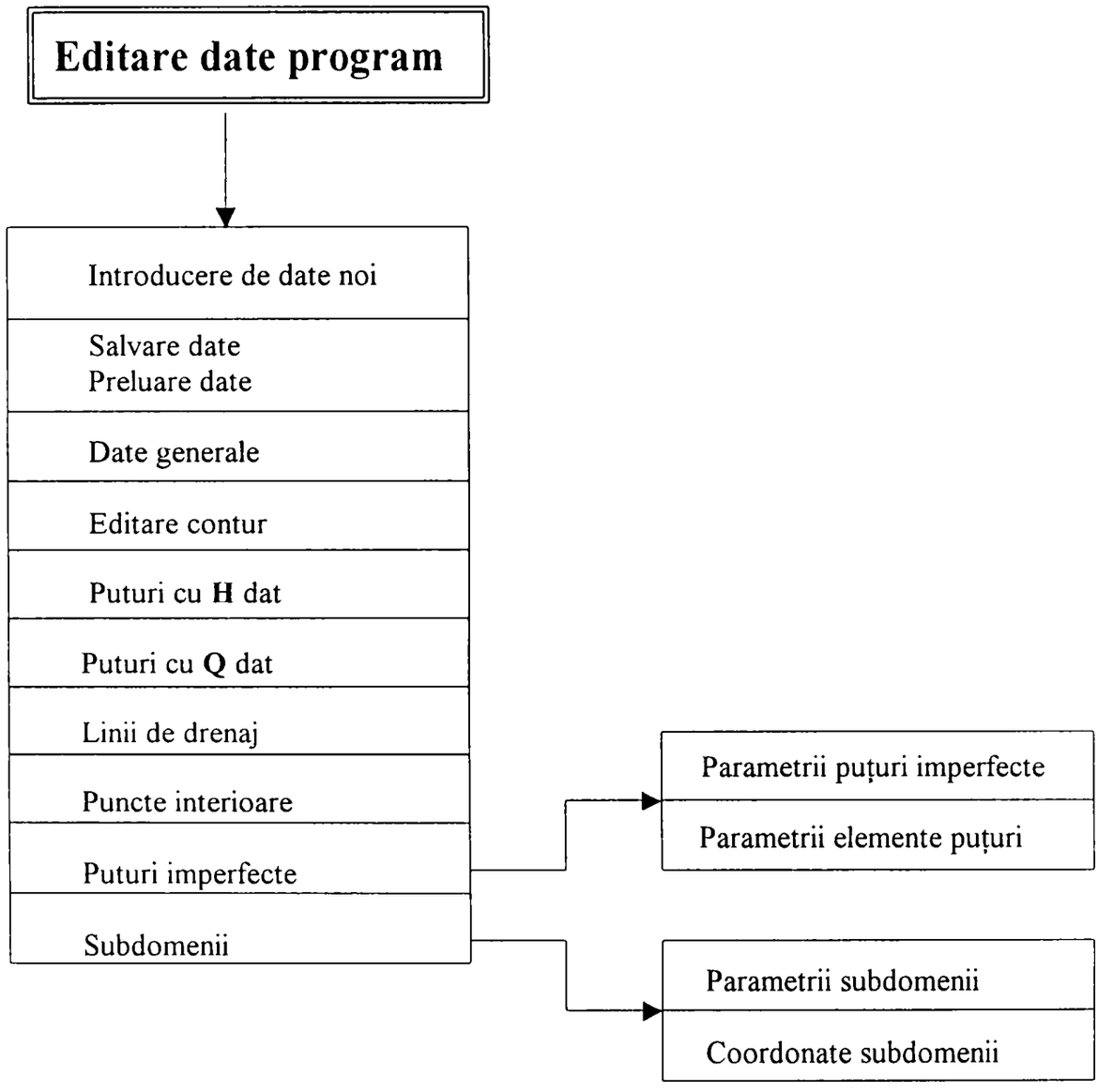
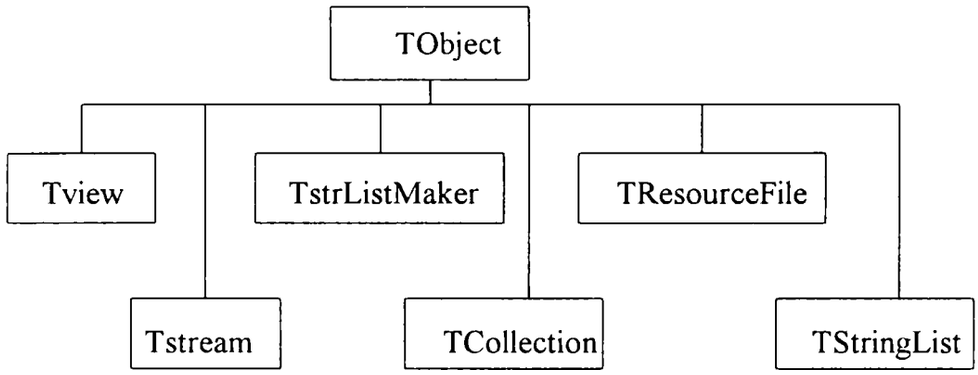


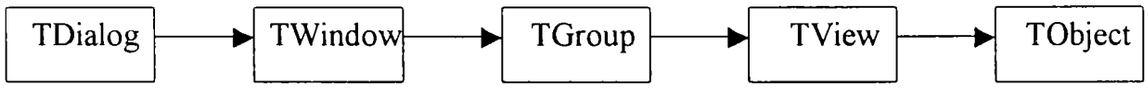
Fig. 7.2 Meniului de editare al datelor

Înainte de a trece la tratarea modului de definre a obiectului ***TBox*** trebuie menționat faptul că ***TObject*** este obiectul de bază al ierarhiei TurboVision. În afară de Tpoint şi Trect toate obiectele standard Turbo Vision derivă din ***TObject***. La primul nivel al ierarhiei sunt obiectele menționate în arborele următor:



Dintre aceste obiecte ***Tview*** este un obiect fundamental pentru Turbo Vision. Orice obiect vizibil trebuie să descindă din ***Tview***. El defineşte o secţiune sau un dreptunghi al ecranului. Totodată gestionează şi orice eveniment care îi este adresat prin metoda ***HandleEvent***.

Aşa după cum se poate observa din definiţia sa, obiectul ***TBox*** moşteneşte toate proprietăţile obiectului ***TDialog***. Privit ca lanţ al moştenirii obiectul ***TDialog*** are următoarea formă:



Revenind la cămpurile obiectului ***TBox*** constatăm că el mai conţine câmpurile TipNod, Abscisa, Ordonata, Potential de tip ***PinputLine***. Pentru introducerea valorilor acestor câmpuri folosindu-se obiecte de tip ***TInputLine***, derivate din ***TView***, al cărui constructor crează o linie în care se pot înscrie un anumit număr de caractere. Pentru etichetarea fiecărei linii cu un text explicativ se utilizează obiectul ***TLabel***, derivat din ***TStaticText***, al cărui constructor crează o etichetă, pusă în evidenţă pe ecranul monitorului în momentul selectării vederii TipNod.

```
R.Assign(16, 2, 28, 3);
TipNod := New(PInputLine, Init(R, 8));
Insert(TipNod);
R.Assign(4, 2, 14, 3);
Insert(New(PLabel, Init(R, '~T~ip Nod:', TipNod)));
```

Constructorul ***TBox.Init*** inițializează fereastra de dialog, care se poate deplasa şi închide dimensiunile ei rămânând nemodificabile. În această se fixează butoanele **Memo**, **Anterior**, **Următor**, **Închide**, inițializează valoarea maximă a contorului la o valoare egală cu MaxNod (numărul maxim de noduri) şi valoarea curentă la 1, după care folosind metoda ***Show*** inițializează câmpurile obiectului ***TContur*** cu valorile corespunzătoare numărului curent al nodului si le afişează în cadrul fereastrei.

```
Constructor TBox.Init;
Var
  R: TRect;

begin
  R.Assign(0, 0, 60, 17);
  inherited Init(R, 'Contur Domeniu  '+FileName);
  Options := Options or ofCentered;
  HelpCtx := $F000;

  R.Assign(16, 2, 28, 3);
  TipNod := New(PInputLine, Init(R, 8));
  Insert(TipNod);
  R.Assign(4, 2, 14, 3);
  Insert(New(PLabel, Init(R, '~T~ip Nod:', TipNod)));

  R.Assign(16, 4, 28, 5);
  Abscisa := New(PInputLine, Init(R, 8));
  Insert(Abscisa);
  R.Assign(4, 4, 14, 5);
  Insert(New(PLabel, Init(R, '~A~bscisa:', Abscisa)));

  R.Assign(16, 6, 28, 7);
  Ordonata := New(PInputLine, Init(R, 8));
  Insert(Ordonata);
  R.Assign(4, 6, 14, 7);
  Insert(New(PLabel, Init(R, '~O~rdonata:', Ordonata)));


  R.Assign(16, 8, 28, 9);
  Potential := New(PInputLine, Init(R, 8));
  Insert(Potential);
  R.Assign(4, 8, 14, 9);
  Insert(New(PLabel, Init(R, '~P~otential', Potential)));

  R.Assign(2, 14, 12, 16);
  Insert(New(PButton, Init(R, '~M~emo', cmRetine, bfNormal)));
  R.Assign(20, 14, 32, 16);
  Insert(New(PButton, Init(R, '~A~nterior', cmAnterior, bfNormal)));
  R.Assign(31, 14, 43, 16);
  Insert(New(PButton, Init(R, '~U~rmator', cmUrmator, bfNormal)));
  R.Assign(46, 14, 57, 16);
  Insert(New(PButton, Init(R, '~T~erminat', cmCancel, bfNormal)));
```

```
  CurrentOrder := 1;
  R.Assign(5, 16, 20, 17);
  Counter := New(PCountView, Init(R));
  with Counter^ do
  begin
    SetCount(MaxNod);SetCurrent(CurrentOrder);
  end;
  Insert(Counter);
  SelectNext(False);
  DisableCommands([cmAnterior]);
  Tct:=Pct;
  Show;
End;
```

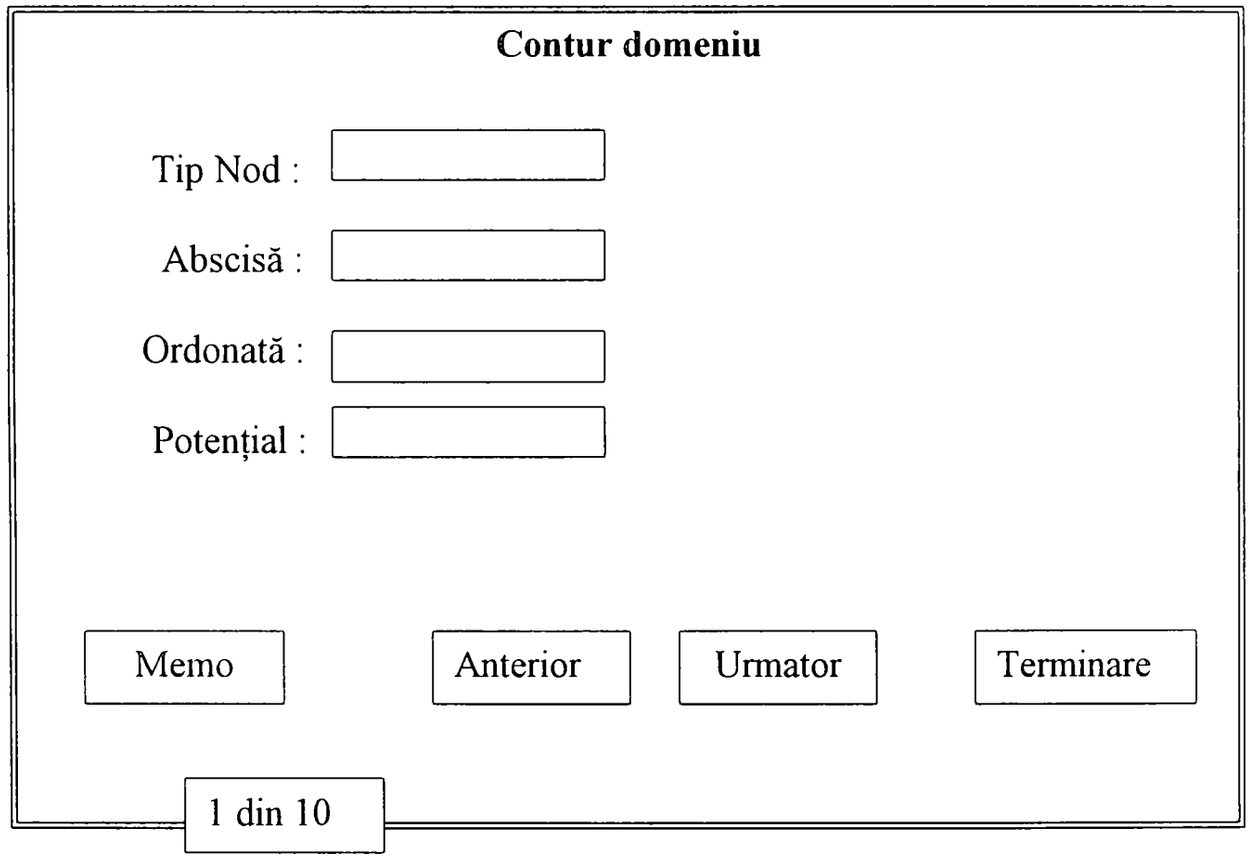


Fig.7.3 Meniul de editare al datelor pentru conturul domeniului

Metoda ***TBox.Show*** , după cum s-a menționat anterior, are funcția de a inițializa câmpurile obiectului ***TBox*** cu valorile corespunzătoare nodului curent. Prin procedura ***SerData*** moştenită de la ***Tview***, se copiază un număr de octeți din parametru în structura internă a obiectului, în cazul de față fereastra de dialog. În plus metoda activează sau dezactivează comenzile ***cmUrmător***, ***cmAnterior*** funcție de valoarea CurrentOrder.

```
Procedure TBox.Show;

Begin
   If CurrentOrder >= MaxNod Then DisableCommands([cmUrmator])
   Else EnableCommands([cmUrmator]);
   If CurrentOrder <=1 Then DisableCommands([cmAnterior])
   Else EnableCommands([cmAnterior]);

   New(PCon);
   With PCon^ Do
     Begin
       TipNod   :=Tct^.TipNod;
       Abscisa  :=Tct^.Abscisa;
       Ordonata :=Tct^.Ordonata;
       Potential:=Tct^.Potential;
     end;
   SetData(PCon^);
end;
```

Metoda ***TBox.Retine*** are funcția de a prelua folosind metoda ***GetData*** datele care se află în fereastra de dialog şi de a inițializa câmpurile obiectului TBox cu aceste valori.

```
Procedure TBox.Retine;

Begin
  GetData(PCon^);
  With Tct^ Do
   Begin
     TipNod   :=PCon^.TipNod;
     Abscisa  :=PCon^.Abscisa;
     Ordonata :=PCon^.Ordonata;
     Potential:=PCon^.Potential;
   end;
end;
```

Destructorul ***TBox.Distruge***, realizează de fapt eliberarea zonei de memorie care a fost ocupată de către lista care a conținut frontiera domeniului. Variabilele dinamice referite se distrug prin apelul procedurii **Dispose**. Practic Dispose, realizează eliberarea efectivă a spațiului de memorie ocupat. Valoarea variabilei Qct trebuie să fie în momentul execuției procedurii Dispose o valoare care a rezultat dintr-un apel al procedurii **New**, reprezentând adresa zonei de memorie care se eliberează. Tipul variabilei utilizate pentru apelul procedurii Dispose trebuie să corespundă cu tipul variabilei utilizate în apelul pereche al procedurii New. În acest mod dimensiunea spațiului de memorie eliberat va fi aceeaşi cu cea a spațiului de memorie alocat.

Zona de memorie Heap este finită, iar modul său de ocupare variază pe parcursul execuției programului, reducându-se prin apelul procedurii New şi crescând prin apelul procedurii Dispose.

Ca şi observație utilizatorul poate afla, prin intermediul funcțiilor predefinite MemAvail şi MaxAvail, care este numărul total de octeți disponibili în zona Heap şi respectiv, în zona liberă de dimensiune maximă.

```
Destructor TBox.Distruge;
Begin
 Qct:=Pct;
 While Pct <> nil Do
  Begin
   Pct:=Qct^.urm;
   Dispose(Qct);
   Qct:=Pct;
  end;
end;
```

Metoda ***HandleEvent*** gestionează evenimentele asociate liniei de stare, prin apelul metodei ***Tview.HandleEvent***, apoi prin testul producerii a trei tipuri speciale de evenimente (mouse, key, broadcast).

```
Procedure TBox.HandleEvent(var Event: TEvent);
Var
  R: TRect;
Begin
  inherited HandleEvent(Event);
  if Event.What = evCommand then
  begin
    case Event.Command of
      cmRetine:
        Begin
          If CurrentOrder > MaxNod Then
            DisableCommands([cmUrmator])
          else
           Begin
            EnableCommands([cmAnterior]);
            Retine;
            If CurrentOrder = MaxNod Then
             Begin
              CurrentOrder:=1;
              Counter^.SetCurrent(CurrentOrder);
              Tct:=Pct;
             end
            else
             Begin
```

```
              CurrentOrder:=CurrentOrder+1;
              Counter^.SetCurrent(CurrentOrder);
              Tct:=Tct^.urm;
             end;
            Show;{Dispose(PCon);}
            ClearEvent(Event);
           end;
      end;


    cmAnterior:
      Begin
        CurrentOrder:=CurrentOrder-1;
        Counter^.SetCurrent(CurrentOrder);
        Tct:=Tct^.ant;Show;Dispose(PCon);
        ClearEvent(Event);
      end;

    cmUrmator:
      Begin
        CurrentOrder:=CurrentOrder+1;
        Counter^.SetCurrent(CurrentOrder);
        Tct:=Tct^.urm; Show;Dispose(PCon);
        ClearEvent(Event);
      end;

end;end;end;
```

Analog acestui procedeu folosit în cadrul submeniului de editare a conturului frontierei sunt construite şi celelalte submeniuri.

7.1.3 Submeniul; Puțuri cu potențial cunoscut.

Acest submeniu este folosit pentru editarea (introducerea, vizualizarea şi/sau modificarea) datelor pentru puțuri cu potențial cunoscut. Utilizatorului i se crează în acest sens facilitatea de a edita următoarele date:

- Coordonatele centrului puțului (abscisă, ordonată);
- Raza puțului;
- Potențialul puțului.

Procedurile şi funcțiile folosite pentru editarea din cadrul acestui submeniu se afla în unitul ***PutHDat.pas***. În cadrul acestui unit puțurile sunt definite ca fiind obiecte de tipul ***TPutH*** în care X, Y, Raza şi Pot sunt câmpurile sale.

Totodată obiectul ***TPutH*** mai conține în plus şi două elemente ***ant*** şi ***urm*** care sunt de tip referință la obiectul ***TPutH***. Deoarece mulțimea puțurilor care alcătuiesc conturul domeniului se află într-o structură de tip listă, elementele de tip referință ***ant*** şi ***urm*** sunt folosite pentru deplasarea şi poziționarea pe un anumit puț din cadrul listei.

Pentru a putea vizualiza câmpurile obiectului ***TPutH*** tot în cadrul unitului ***PutHDat.pas*** este definit obiectul ***TPutHBox***.

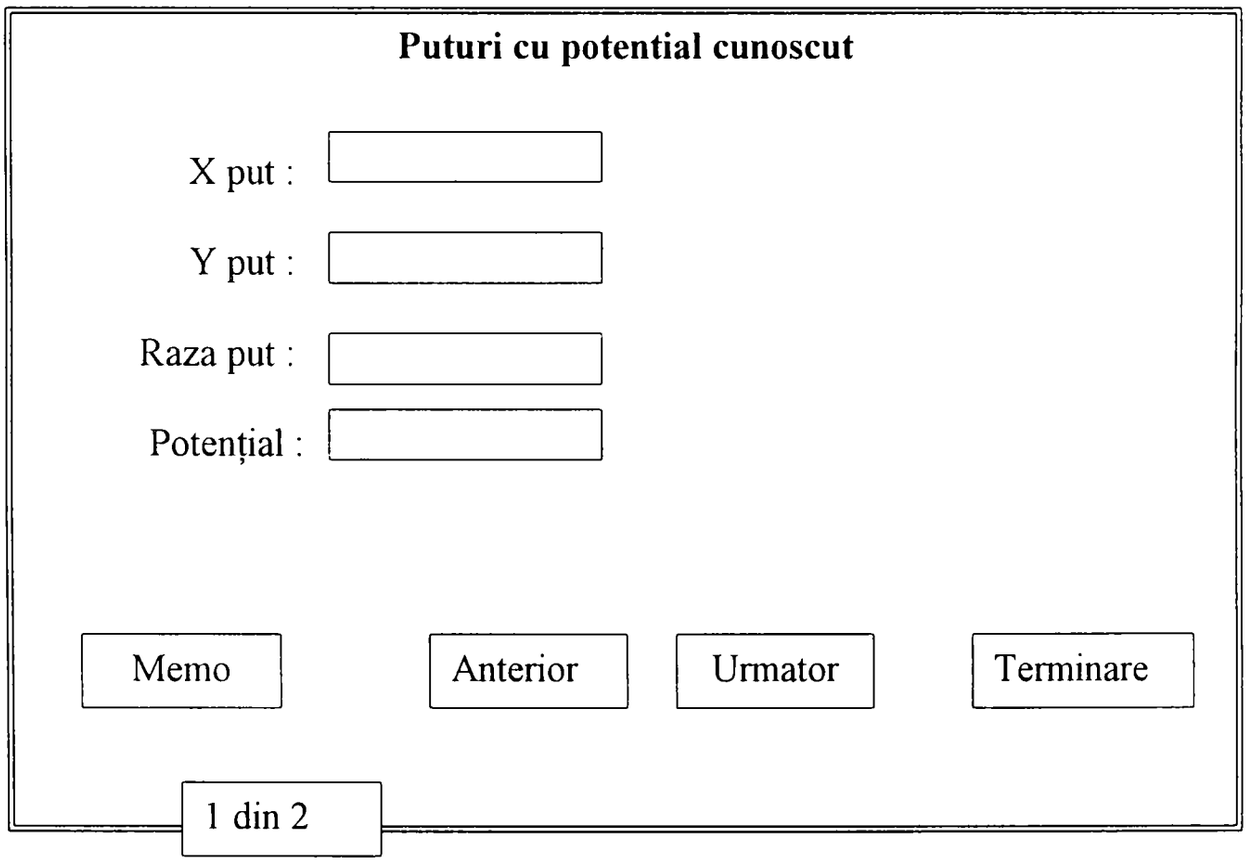


Fig. 7.4 Meniul de editare al datelor pentru puțuri cu potențial cunoscut

```
PPutHBox = ^TPutHBox;
 TPutHBox = Object(TDialog)
 X,Y,Raza,Pot:PInputLine;
 Counter:PCountView;
 Constructor Init;
 Procedure HandleEvent(var Event: TEvent);virtual;
 Procedure Retine;
 Procedure Show;
end;
```

Câmpurile obiectului ***TPutHBox*** sunt X, Y, Raza, Pot de tip ***PinputLine***. Pentru introducerea valorilor acestor câmpuri se folosesc obiecte de tip ***TInputLine***, derivate din ***TView***, al cărui constructor crează o linie în care se pot înscrie un anumit număr de caractere. Pentru etichetarea fiecărei linii cu un text explicativ se utilizează obiectul ***TLabel***, derivat din ***TStaticText***, al cărui constructor crează o etichetă, pusă în evidență pe ecranul monitorului în momentul selectării unuia din câmpuri. În continuare este exemplificat modul de realizare al ferestrei pentru editarea valorilor în cazul puțurilor cu potențial cunoscut.

```
R.Assign(16, 2, 28, 3);
X := New(PInputLine, Init(R, 8));
Insert(X);
R.Assign(4, 2, 14, 3);
Insert(New(PLabel, Init(R, '~X~ Put:', X)));

R.Assign(16, 4, 28, 5);
Y := New(PInputLine, Init(R, 8));
Insert(Y);
R.Assign(4, 4, 14, 5);
Insert(New(PLabel, Init(R, '~Y~ Put:', Y)));

R.Assign(16, 6, 28, 7);
Raza := New(PInputLine, Init(R, 8));
Insert(Raza);
R.Assign(4, 6, 14, 7);
Insert(New(PLabel, Init(R, '~R~aza:', Raza)));

R.Assign(16, 8, 28, 9);
Pot := New(PInputLine, Init(R, 8));
Insert(Pot);
R.Assign(4, 8, 14, 9);
Insert(New(PLabel, Init(R, '~P~otential', Pot)));
```

Constructorul ***TPutHBox.Init*** inițializează de fapt fereastra de dialog, care se poate deplasa şi închide dimensiunile ei rămânând nemodificabile. În această fereastră se fixează şi butoanele **Memo**, **Anterior**, **Următor** şi **Închide**, se inițializează valoarea maximă a contorului la o valoare egală cu MaxH (numărul maxim de puțuri) şi valoarea curentă la 1, după care folosind metoda ***Show*** inițializează câmpurile obiectului ***TPutH*** cu valorile corespunzătoare numărului curent al puțului cu potențial cunoscut şi le afişează în cadrul ferestrei.

```
Constructor TPutHBox.init;

var
  R: TRect;

begin
  R.Assign(0, 0, 60, 17);
  inherited Init(R, 'Puturi cu potential cunoscut');
  Options := Options or ofCentered;
  HelpCtx := $F000;
```

```
  R.Assign(16, 2, 28, 3);
  X := New(PInputLine, Init(R, 8));
  Insert(X);
  R.Assign(4, 2, 14, 3);
  Insert(New(PLabel, Init(R, '~X~ Put:', X)));

  R.Assign(16, 4, 28, 5);
  Y := New(PInputLine, Init(R, 8));
  Insert(Y);
  R.Assign(4, 4, 14, 5);
  Insert(New(PLabel, Init(R, '~Y~ Put:', Y)));
  R.Assign(16, 6, 28, 7);
  Raza := New(PInputLine, Init(R, 8));
  Insert(Raza);
  R.Assign(4, 6, 14, 7);
  Insert(New(PLabel, Init(R, '~R~aza:', Raza)));

  R.Assign(16, 8, 28, 9);
  Pot := New(PInputLine, Init(R, 8));
  Insert(Pot);
  R.Assign(4, 8, 14, 9);
  Insert(New(PLabel, Init(R, '~P~otential', Pot)));

  R.Assign(2, 14, 12, 16);
  Insert(New(PButton, Init(R, '~O~k', cmRetine, bfNormal)));
  R.Assign(20, 14, 32, 16);
  Insert(New(PButton, Init(R, '~A~nterior', cmAnterior, bfNormal)));
  R.Assign(31, 14, 43, 16);
  Insert(New(PButton, Init(R, '~U~rmator', cmUrmator, bfNormal)));
  R.Assign(46, 14, 57, 16);
  Insert(New(PButton, Init(R, '~A~nulare', cmCancel, bfNormal)));

  CurrentOrder := 1;
  R.Assign(5, 16, 20, 17);
  Counter := New(PCountView, Init(R));
  with Counter^ do
  begin
    SetCount(MaxH);
    SetCurrent(CurrentOrder);
  end;

  Insert(Counter);
  SelectNext(False);
  DisableCommands([cmAnterior]);
  Th:=Ph;
  Show;
end;
```

Metoda ***TPutHBox.Show*** , după cum s-a menționat anterior, are funcția de a inițializa câmpurile obiectului ***TPutH*** cu valorile corespunzătoare puțului curent (puț cu potențial cunoscut). Prin procedura ***SetData*** moştenită de la ***Tview***, se copiază un număr de octeți din parametru în structura internă a obiectului, în cazul de față fereastra de dialog. În plus metoda activează sau dezactivează comenzile ***cmUrmător***, ***cmAnterior*** funcție de valoarea lui CurrentOrder.

```
Procedure TPutHBox.Show;

Begin
   If CurrentOrder >= MaxH Then
      DisableCommands([cmUrmator])
   else
      EnableCommands([cmUrmator]);
   If CurrentOrder <=1 Then
      DisableCommands([cmAnterior])
   else
      EnableCommands([cmAnterior]);
   New(PutH);
   With PutH^ Do
     Begin
      X:=Th^.X;
      Y:=Th^.Y;
      Raza:=Th^.Raza;
      Pot:=Th^.Pot;
     end;
   SetData(PutH^);
end;
```

Metoda ***TPutHBox.Retine*** are funcția de a prelua folosind metoda ***GetData*** datele care se află în fereastra de dialog şi de a inițializa câmpurile obiectului TBox cu aceste valori.

```
Procedure TPutHBox.Retine;

Begin
  GetData(PutH^);
  With Th^ Do
   Begin
     X    :=PutH^.X;
     Y    :=PutH^.Y;
     Raza :=PutH^.Raza;
     Pot  :=PutH^.Pot;
   end;
end;
```

Destructorul ***TPutHBox.Distruge***, realizează de fapt eliberarea zonei de memorie care a fost ocupată de către lista care a conținut frontiera domeniului. Variabilele dinamice referite se distrug prin apelul procedurii **Dispose**.

```
Destructor TPutHBox.Distruge;
Begin
 Qh:=Ph;
 While Ph <> nil Do
  Begin
   Ph:=Qh^.urm;
   Dispose(Qh);
   Qh:=Ph;
  end;
end;
```

Metoda ***TputHBox.HandleEvent*** gestionează evenimentele asociate liniei de stare, prin apelul metodei ***Tview.HandleEvent***, apoi prin testul producerii a trei tipuri speciale de evenimente (mouse, key, broadcast).

```
Procedure TPutHBox.HandleEvent(var Event: TEvent);

Var  R: TRect;
Begin
  inherited HandleEvent(Event);
  if Event.What = evCommand then
  begin
    Case Event.Command of
      cmRetine:
       Begin
        If CurrentOrder > MaxH Then
         Begin
          DisableCommands([cmUrmator])
         end
        else
         Begin
          EnableCommands([cmAnterior]);
          Retine;
          If CurrentOrder = MaxH Then
            Begin
             CurrentOrder:=1;
             Counter^.SetCurrent(CurrentOrder);
             Th:=Ph;
            end
          else
            Begin
             CurrentOrder:=CurrentOrder+1;
             Counter^.SetCurrent(CurrentOrder);
             Th:=Th^.urm;
            end;
         Show;Dispose(PutH);
         ClearEvent(Event);
        end;
       end;
      cmAnterior:
        Begin
          CurrentOrder:=CurrentOrder-1;
          Counter^.SetCurrent(CurrentOrder);
          Th:=Th^.ant;
          Show;Dispose(PutH);
          ClearEvent(Event);
        end;
      cmUrmator:
        Begin
          CurrentOrder:=CurrentOrder+1;
          Counter^.SetCurrent(CurrentOrder);
          Th:=Th^.urm;
          Show;Dispose(PutH);
          ClearEvent(Event);
        end;  end; end;end;
```

7.1.4 Submeniul; Puțuri cu flux cunoscut.

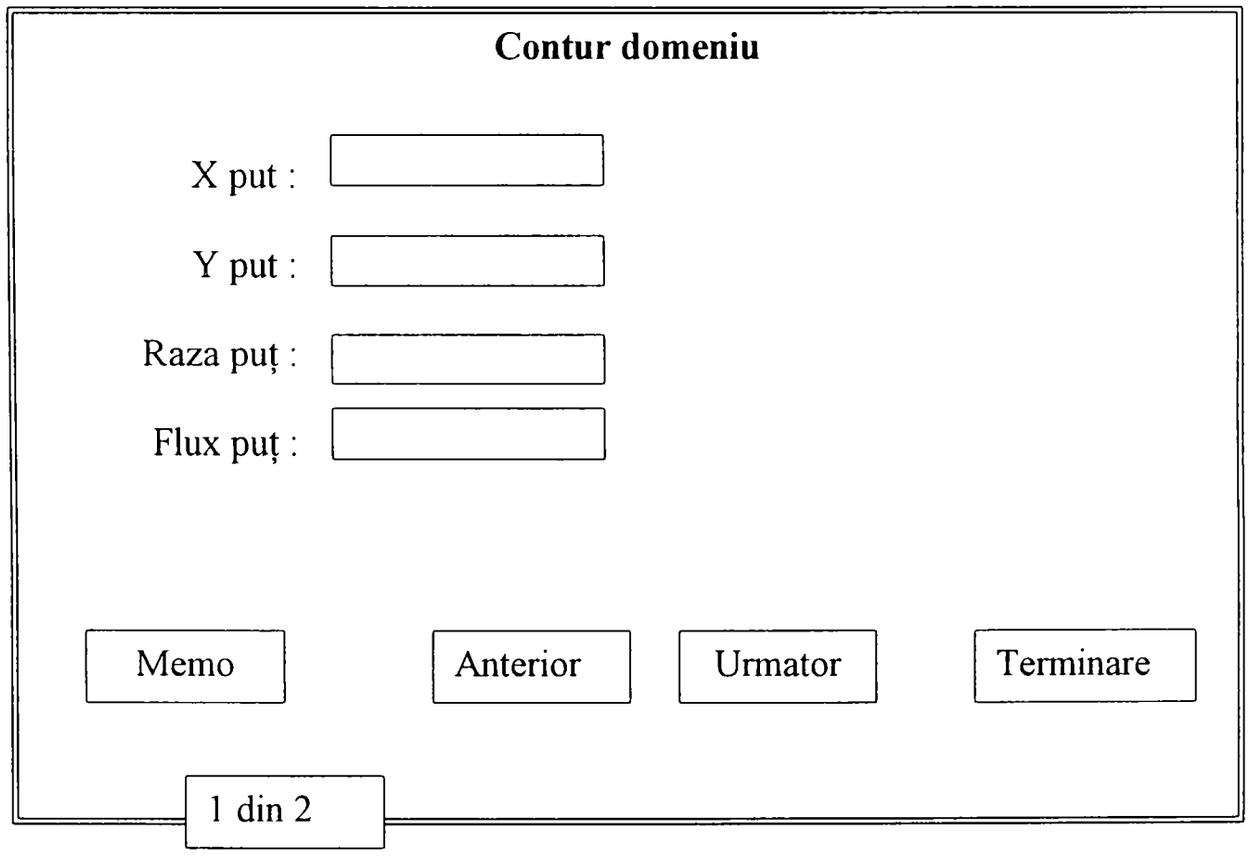
Acest submeniu este folosit pentru editarea (introducerea, vizualizarea şi/sau modificarea) datelor pentru puțuri cu flux cunoscut. Utilizatorului i se crează în acest sens facilitatea de a edita următoarele date:

- Coordonatele centrului puțului (abscisă, ordonată);
- Raza puțului;
- Debitul (fluxul) puțului.

Procedurile şi funcțiile folosite pentru editarea din cadrul acestui submeniu se afla în unitul ***PutQDat.pas***. În cadrul acestui unit puțurile sunt definite ca fiind obiecte de tipul ***TPutQ*** în care X, Y, Raza şi Pot sunt câmpurile sale.

Totodată obiectul ***TPutQ*** mai conține în plus şi elementele ***ant*** şi ***urm*** care sunt de tip referință la obiectul ***TPutQ***. Şi în acest caz, deoarece mulțimea puțurilor care alcătuiesc conturul domeniului se află într-o structură de tip listă, elementele de tip referință ***ant*** şi ***urm*** sunt folosite pentru deplasarea şi poziționarea pe un anumit puț din cadrul listei.

Pentru a putea vizualiza câmpurile obiectului ***TPutQ*** tot în cadrul unitului ***PutQDat.pas*** este definit obiectul ***TPutQBox***.



```
PPutQBox = ^TPutQBox;
   TPutQBox = Object(TDialog)
   X,Y,Raza,Pot:PInputLine;
   Counter:PCountView;
   Constructor Init;
   Procedure HandleEvent(var Event: TEvent);virtual;
   Procedure Show;
   Procedure Retine;
  end;
```

Câmpurile obiectului ***TPutQBox*** sunt X, Y, Raza, Pot de tip ***PinputLine***. Pentru introducerea valorilor acestor câmpuri se folosesc obiecte de tip ***TInputLine***, derivate din ***TView***, al cărui constructor crează o linie în care se pot înscrie un anumit număr de caractere. Şi în acest caz pentru etichetarea fiecărei linii cu un text explicativ se utilizează obiectul ***TLabel***, derivat din ***TStaticText***, al cărui constructor crează o etichetă, pusă în evidenţă pe ecranul monitorului în momentul selectării unuia din câmpuri. În continuare este exemplificat modul de realizare al ferestrei pentru editarea valorilor în cazul puţurilor cu flux cunoscut.

```
R.Assign(16, 2, 28, 3);
X := New(PInputLine, Init(R, 8));
Insert(X);
R.Assign(4, 2, 14, 3);
Insert(New(PLabel, Init(R, '~X~ Put:', X)));

R.Assign(16, 4, 28, 5);
Y := New(PInputLine, Init(R, 8));
Insert(Y);
R.Assign(4, 4, 14, 5);
Insert(New(PLabel, Init(R, '~Y~ Put:', Y)));

R.Assign(16, 6, 28, 7);
Raza := New(PInputLine, Init(R, 8));
Insert(Raza);
R.Assign(4, 6, 14, 7);
Insert(New(PLabel, Init(R, '~R~aza:', Raza)));

R.Assign(16, 8, 28, 9);
Pot := New(PInputLine, Init(R, 8));
Insert(Pot);
R.Assign(4, 8, 14, 9);
Insert(New(PLabel, Init(R, '~F~lux', Pot)));
```

Constructorul ***TPutQBox.Init*** iniţializează de fapt fereastra de dialog, care se poate deplasa şi închide dimensiunile ei rămânând nemodificabile. În această fereastră se fixează şi butoanele **Memo**, **Anterior**, **Următor** şi **Închide**, se iniţializează valoarea maximă a contorului la o valoare egală cu MaxH (numărul maxim de puţuri) şi valoarea curentă la 1, după care folosind metoda ***Show*** iniţializează câmpurile obiectului ***TPutQ*** cu valorile corespunzătoare numărului curent al puţului cu potenţial cunoscut şi le afişează în cadrul fereastrei.

```
Constructor TPutQBox.init;

var
  R: TRect;

begin

  R.Assign(0, 0, 60, 17);
  inherited Init(R, 'Puturi cu flux cunoscut');
  Options := Options or ofCentered;
  HelpCtx := $F000;

  R.Assign(16, 2, 28, 3);
  X := New(PInputLine, Init(R, 8));
  Insert(X);
  R.Assign(4, 2, 14, 3);
  Insert(New(PLabel, Init(R, '~X~ Put:', X)));

  R.Assign(16, 4, 28, 5);
  Y := New(PInputLine, Init(R, 8));
  Insert(Y);
  R.Assign(4, 4, 14, 5);
  Insert(New(PLabel, Init(R, '~Y~ Put:', Y)));

  R.Assign(16, 6, 28, 7);
  Raza := New(PInputLine, Init(R, 8));
  Insert(Raza);
  R.Assign(4, 6, 14, 7);
  Insert(New(PLabel, Init(R, '~R~aza:', Raza)));

  R.Assign(16, 8, 28, 9);
  Pot := New(PInputLine, Init(R, 8));
  Insert(Pot);
  R.Assign(4, 8, 14, 9);
  Insert(New(PLabel, Init(R, '~F~lux', Pot)));



  R.Assign(2, 14, 12, 16);
  Insert(New(PButton, Init(R, '~O~k', cmRetine, bfNormal)));
  R.Assign(20, 14, 32, 16);
  Insert(New(PButton, Init(R, '~A~nterior', cmAnterior, bfNormal)));
  R.Assign(31, 14, 43, 16);
  Insert(New(PButton, Init(R, '~U~rmator', cmUrmator, bfNormal)));
  R.Assign(46, 14, 57, 16);
  Insert(New(PButton, Init(R, '~A~nulare', cmCancel, bfNormal)));

  CurrentOrder := 1;

  R.Assign(5, 16, 20, 17);
  Counter := New(PCountView, Init(R));
  with Counter^ do
  begin
    SetCount(MaxQ);
    SetCurrent(CurrentOrder);
  end;
```

```
  Insert(Counter);

  SelectNext(False);

  DisableCommands([cmAnterior]);
  Tq:=Pq;
  Show;
end;
```

Metoda ***TPutQBox.Show*** , după cum s-a menționat anterior, are funcția de a inițializa câmpurile obiectului ***TPutQ*** cu valorile corespunzătoare puțului curent (puț cu potențial cunoscut). Prin procedura ***SetData*** moștenită de la ***Tview***, se copiază un număr de octeți din parametru în structura internă a obiectului, în cazul de față fereastra de dialog. În plus metoda activează sau dezactivează comenzile ***cmUrmător***, ***cmAnterior*** funcție de valoarea lui CurrentOrder care indică poziția puțului curent.

```
Procedure TPutQBox.Show;

Begin
   If CurrentOrder >= MaxQ Then
      DisableCommands([cmUrmator])
   else
      EnableCommands([cmUrmator]);
   If CurrentOrder <=1 Then
      DisableCommands([cmAnterior])
   else
      EnableCommands([cmAnterior]);
   New(PutQ);
   With PutQ^ Do
     Begin
      X:=Tq^.X;
      Y:=Tq^.Y;
      Raza:=Tq^.Raza;
      Pot:=Tq^.Pot;
     end;
   SetData(PutQ^);
end;
```

Metoda ***TPutQBox.Retine*** are funcția de a prelua folosind metoda ***GetData*** datele care se află în fereastra de dialog şi de a inițializa câmpurile obiectului TBox cu aceste valori.

```
Procedure TPutQBox.Retine;

Begin
  GetData(PutQ^);
  With Tq^ Do
   Begin
     X:=PutQ^.X;
     Y:=PutQ^.Y;
     Raza:=PutQ^.Raza;
     Pot:=PutQ^.Pot;
   end;
end;
```

Destructorul ***TPutQBox.Distruge***, realizează de fapt eliberarea zonei de memorie care a fost ocupată de către lista care a conținut frontiera domeniului. Variabilele dinamice referite se distrug prin apelul procedurii **Dispose**.

```
Destructor TPutQBox.Distruge;
Begin
 Qq:=Pq;
 While Pq <> nil Do
  Begin
   Pq:=Qq^.urm;
   Dispose(Qq);
   Qq:=Pq;
  end;
end;
```

Metoda ***TputQBox.HandleEvent*** gestionează evenimentele asociate liniei de stare, prin apelul metodei ***Tview.HandleEvent***, apoi prin testul producerii a trei tipuri speciale de evenimente (mouse, key, broadcast).

```
procedure TPutQBox.HandleEvent(var Event: TEvent);
var
  R: TRect;
begin
  inherited HandleEvent(Event);
  if Event.What = evCommand then
  begin
    case Event.Command of
      cmRetine:
        Begin
           If CurrentOrder > MaxQ Then
           Begin
            DisableCommands([cmUrmator])
           end
          else
           Begin
            EnableCommands([cmAnterior]);
            Retine;
            If CurrentOrder = MaxQ Then
```

```
                Begin
                 CurrentOrder:=1;
                 Counter^.SetCurrent(CurrentOrder);
                 Tq:=Pq;
                end
              else
                Begin
                 CurrentOrder:=CurrentOrder+1;
                 Counter^.SetCurrent(CurrentOrder);
                 Tq:=Tq^.urm;
                end;
            Show;Dispose(PutQ);
            ClearEvent(Event);
            end;
          end;
        cmAnterior:
          Begin
            CurrentOrder:=CurrentOrder-1;
            Counter^.SetCurrent(CurrentOrder);
            Tq:=Tq^.ant;
            Show;Dispose(PutQ);
            ClearEvent(Event);
          end;
        cmUrmator:
          Begin
            CurrentOrder:=CurrentOrder+1;
            Counter^.SetCurrent(CurrentOrder);
            Tq:=Tq^.urm;
            Show;Dispose(PutQ);
            ClearEvent(Event);
          end;
    end;
   end;
  end;
```

7.1.5 Submeniul; Linii de drenaj.

Acest submeniu este folosit pentru editarea (introducerea, vizualizarea şi/sau modificarea) datelor pentru liniile de drenaj. Utilizatorului i se crează în acest sens facilitatea de a edita următoarele date:

- Coordonatele nodurilor care alcătuiesc linia de drenaj (abscisă, ordonată);
- Potențialul liniei de dren;
- Parametrii imperfecțiunii (P_1 , P_2 , P_3);
- Tipul elementului.

```
{------------------------------------------------------------------------}
{                                                                        }
{    Precizări asupra  parametrilor imperfecțiunii funcție de           }
{                        tipul nodului.                                  }
{                                                                        }
{------------------------------------------------------------------------}
{                                                                        }
{   Noduri de tip 5                  :    Ecran parțial penetrat        }
{   Parametrii imperfecțiunii        :    P1 - Pătrundere ecran         }
{                                    :    P2 - Grosime strat            }
{                                    :    P3 - Nu exista                }
{------------------------------------------------------------------------}
{                                                                        }
{   Noduri de tip 6                  :    Infiltrații prin radier plan  }
{   Parametrii imperfecțiunii        :    P1 - Lățime radier            }
{                                    :    P2 - Adâncime strat acvifer   }
{                                    :    P3 - Nu exista                }
{------------------------------------------------------------------------}
{                                                                        }
{   Noduri de tip 7                  :  Infiltrații prin tub de drenaj  }
{   Parametrii imperfecțiunii        :    P1 - Diametrul tub drenaj     }
{                                    :    P2 - Grosime strat            }
{                                    :    P3 - Adâncime de pozare       }
{------------------------------------------------------------------------}
{                                                                        }
{   Noduri de tip 8                  : Infiltrații prin radier          }
{                                       curbiliniu                       }
{   Parametrii imperfecțiunii :      :    P1 - Lățime canal             }
{                                    :    P2 - Adâncime strat           }
{                                    :    P3 - Adâncime de pătrundere   }
{                                       canal                            }
{------------------------------------------------------------------------}
```

Procedurile şi funcțiile folosite pentru editarea din cadrul acestui submeniu se afla în unitul ***Drenuri.pas***. În cadrul acestui unit puțurile sunt definite ca fiind obiecte de tipul ***TDrenaj*** în care Tip, X, Y, H , P_1 , P_2 , P_3 sunt câmpurile sale.

Totodată obiectul ***TDrenaj*** mai conține în plus şi elementele ***ant*** şi ***urm*** care sunt de tip referință la obiectul ***TDrenaj***. Şi în acest caz, deoarece mulțimea elementelor care alcătuiesc linia de drenaj se află într-o structură de tip listă, elementele de tip referință ***ant*** şi ***urm*** sunt folosite pentru deplasarea şi poziționarea pe un anumit element din cadrul listei.

Pentru a putea vizualiza câmpurile obiectului ***TDrenaj*** tot în cadrul unitului ***Drenajt.pas*** este definit obiectul ***TPDrenBox***.

```
PDrenBox = ^TPDrenBox;
 TPDrenBox = Object(TDialog)
 Tip,X,Y,H,P1,P2,P3 :PInputLine;
 Counter:PCountView;
 Constructor Init;
 Procedure HandleEvent(var Event: TEvent);virtual;
 Procedure Show;
 Procedure Retine;
end;
```

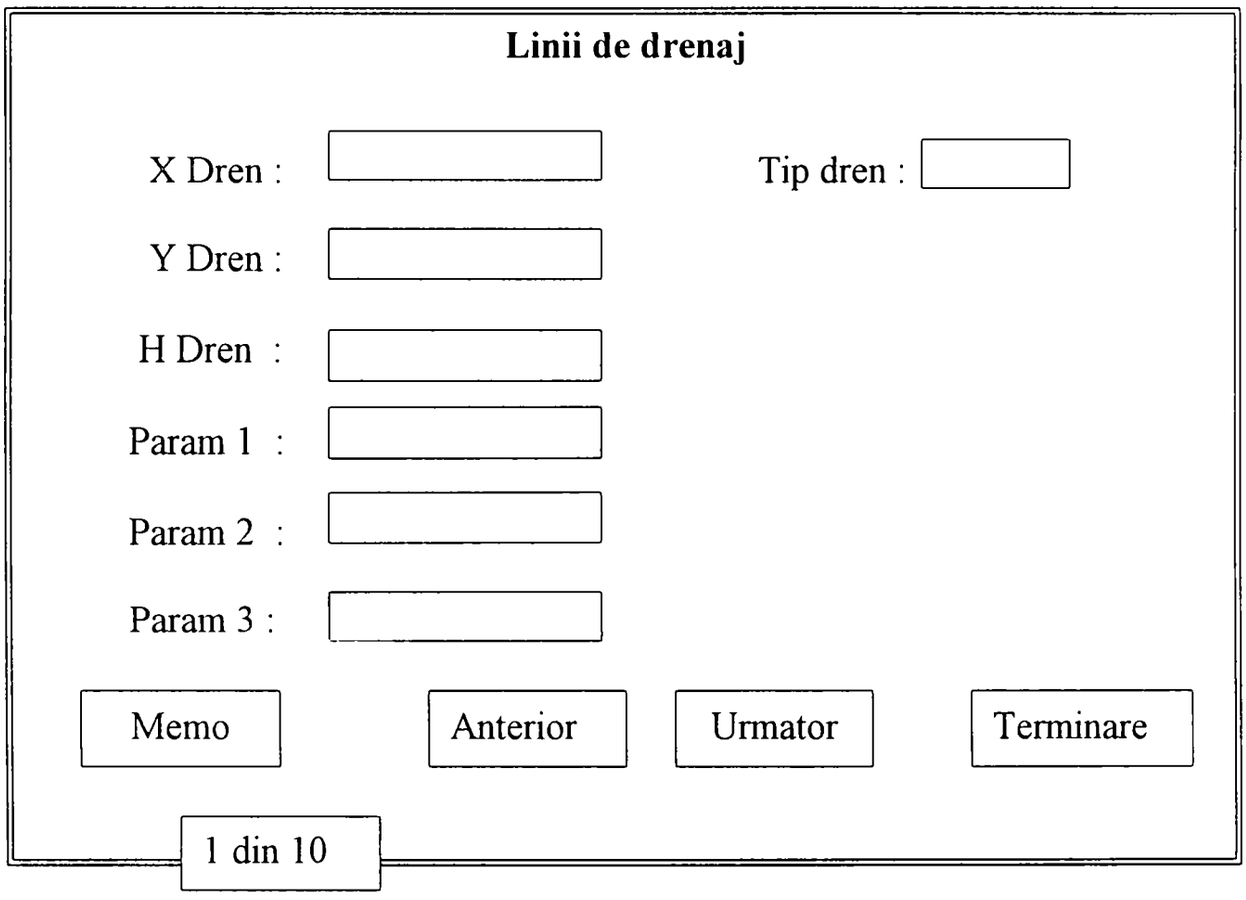


Fig. 7.6 Meniul de editare al datelor pentru liniile de drenaj

Analizând câmpurile obiectului ***TPDrenBox*** constatăm că el conține câmpurile Tip, X, Y, H, P1, P2, P3 de tip ***PinputLine***. Pentru introducerea valorilor acestor câmpuri se folosesc obiecte de tip ***TInputLine***, derivate din ***TView***, al cărui constructor crează o linie în care se pot înscrie un anumit număr de caractere. Şi în acest caz pentru etichetarea fiecărei linii cu un text explicativ se utilizează obiectul ***TLabel***, derivat din ***TStaticText***, al cărui constructor crează o etichetă, pusă în evidență pe ecranul monitorului în momentul selectării unuia din câmpuri. În continuare este exemplificat modul de realizare al ferestrei pentru editarea valorilor în cazul liniilor de drenaj.

```
R.Assign(42, 2, 48, 3);
Tip := New(PInputLine, Init(R, 8));
Insert(Tip);
R.Assign(30, 2, 40, 3);
Insert(New(PLabel, Init(R, '~Tip~ Dren:', Tip)));
```

```
R.Assign(16, 2, 28, 3);
X := New(PInputLine, Init(R, 8));
Insert(X);
R.Assign(4, 2, 14, 3);
Insert(New(PLabel, Init(R, '~X~ Dren:', X)));


R.Assign(16, 4, 28, 5);
Y := New(PInputLine, Init(R, 8));
Insert(Y);
R.Assign(4, 4, 14, 5);
Insert(New(PLabel, Init(R, '~Y~ Dren:', Y)));


R.Assign(16, 6, 28, 7);
H := New(PInputLine, Init(R, 8));
Insert(H);
R.Assign(4, 6, 14, 7);
Insert(New(PLabel, Init(R, '~H~ Dren:', H)));


R.Assign(16, 8, 28, 9);
P1 := New(PInputLine, Init(R, 8));
Insert(P1);
R.Assign(4, 8, 14, 9);
Insert(New(PLabel, Init(R, '~P~aram 1', P1)));


R.Assign(16, 10, 28, 11);
P2 := New(PInputLine, Init(R, 8));
Insert(P2);
R.Assign(4, 10, 14, 11);
Insert(New(PLabel, Init(R, '~P~aram 2', P2)));


R.Assign(16, 12, 28, 13);
P3 := New(PInputLine, Init(R, 8));
Insert(P3);
R.Assign(4, 12, 14, 13);
Insert(New(PLabel, Init(R, '~P~aram 3', P3)));
```

Constructorul ***TPDrenBox.Init*** iniţializează de fapt fereastra de dialog, care se poate deplasa şi închide dimensiunile ei rămânând nemodificabile. În această fereastră se fixează şi butoanele **Memo**, **Anterior**, **Următor** şi **Închide**, se iniţializează valoarea maximă a contorului la o valoare egală cu MaxDren (numărul maxim de noduri ale liniilor de drenaj) şi valoarea curentă la 1, după care folosind metoda ***Show*** iniţializează câmpurile obiectului ***TDrenaj*** cu valorile corespunzătoare numărului curent elementului şi le afişează în cadrul fereastrei.

```
Constructor TPDrenBox.init;

var
  R: TRect;

begin

  R.Assign(0, 0, 60, 17);
  inherited Init(R, 'Linii de drenaj');
  Options := Options or ofCentered;
  HelpCtx := $F000;

  R.Assign(42, 2, 48, 3);
  Tip := New(PInputLine, Init(R, 8));
  Insert(Tip);
  R.Assign(30, 2, 40, 3);
  Insert(New(PLabel, Init(R, '~Tip~ Dren:', Tip)));

  R.Assign(16, 2, 28, 3);
  X := New(PInputLine, Init(R, 8));
  Insert(X);
  R.Assign(4, 2, 14, 3);
  Insert(New(PLabel, Init(R, '~X~ Dren:', X)));

  R.Assign(16, 4, 28, 5);
  Y := New(PInputLine, Init(R, 8));
  Insert(Y);
  R.Assign(4, 4, 14, 5);
  Insert(New(PLabel, Init(R, '~Y~ Dren:', Y)));

  R.Assign(16, 6, 28, 7);
  H := New(PInputLine, Init(R, 8));
  Insert(H);
  R.Assign(4, 6, 14, 7);
  Insert(New(PLabel, Init(R, '~H~ Dren:', H)));

  R.Assign(16, 8, 28, 9);
  P1 := New(PInputLine, Init(R, 8));
  Insert(P1);
  R.Assign(4, 8, 14, 9);
  Insert(New(PLabel, Init(R, '~P~aram 1', P1)));


  R.Assign(16, 10, 28, 11);
  P2 := New(PInputLine, Init(R, 8));
  Insert(P2);
  R.Assign(4, 10, 14, 11);
  Insert(New(PLabel, Init(R, '~P~aram 2', P2)));


  R.Assign(16, 12, 28, 13);
  P3 := New(PInputLine, Init(R, 8));
  Insert(P3);
  R.Assign(4, 12, 14, 13);
  Insert(New(PLabel, Init(R, '~P~aram 3', P3)));
```

```
  R.Assign(2, 14, 12, 16);
  Insert(New(PButton, Init(R, '~M~emo', cmRetine, bfNormal)));
  R.Assign(20, 14, 32, 16);
  Insert(New(PButton, Init(R, '~A~nterior', cmAnterior, bfNormal)));
  R.Assign(31, 14, 43, 16);
  Insert(New(PButton, Init(R, '~U~rmator', cmUrmator, bfNormal)));
  R.Assign(46, 14, 57, 16);
  Insert(New(PButton, Init(R, '~A~nulare', cmCancel, bfNormal)));

  CurrentOrder := 1;
  R.Assign(5, 16, 20, 17);
  Counter := New(PCountView, Init(R));
  with Counter^ do
  begin
    SetCount(MaxDren);
    SetCurrent(CurrentOrder);
  end;
  Insert(Counter);
  SelectNext(False);
  DisableCommands([cmAnterior]);
  Td:=Pd;
  Show;
end;
```

Metoda ***TPDrenBox.Show*** , după cum s-a menționat anterior, are funcția de a inițializa câmpurile obiectului ***TDrenaj*** cu valorile corespunzătoare elementului curent. Prin procedura ***SetData*** moştenită de la ***Tview***, se copiază un număr de octeți din parametru în structura internă a obiectului, în cazul de față fereastra de dialog. În plus metoda activează sau dezactivează comenzile ***cmUrmător***, ***cmAnterior*** funcție de valoarea lui CurrentOrder care indică poziția elementului curent.

```
Procedure TPDrenBox.Show;

Begin
   If CurrentOrder >= MaxDren Then
      DisableCommands([cmUrmator])
   else
      EnableCommands([cmUrmator]);
   If CurrentOrder <=1 Then
      DisableCommands([cmAnterior])
   else
      EnableCommands([cmAnterior]);
   New(Drenaj);
   With Drenaj^ Do
     Begin
      Tip:=Td^.Tip;
      X:=Td^.x;
      Y:=Td^.y;
      H:=Td^.h;
      P1:=Td^.P1;
      P2:=Td^.P2;
      P3:=Td^.P3;
     end;
   SetData(Drenaj^);end;
```

Metoda ***TDrenBox.Retine*** are funcția de a prelua folosind metoda ***GetData*** datele care se află în fereastra de dialog şi de a inițializa câmpurile obiectului TBox cu aceste valori.

```
Procedure TPDrenBox.Retine;

Begin
  GetData(Drenaj^);
  With Td^ Do
   Begin
    Tip  :=Drenaj^.Tip;
    X    :=Drenaj^.X;
    Y    :=Drenaj^.Y;
    H    :=Drenaj^.H;
    P1   :=Drenaj^.P1;
    P2   :=Drenaj^.P2;
    P3   :=Drenaj^.P3;
   end;
end;
```

Destructorul ***TPDrenBox.Distruge***, realizează de fapt eliberarea zonei de memorie care a fost ocupată de către lista care a conținut frontiera domeniului. Variabilele dinamice referite se distrug prin apelul procedurii **Dispose**.

```
Destructor TPDrenBox.Distruge;
Begin
 Qd:=Pd;
 While Pd <> nil Do
  Begin
   Pd:=Qd^.urm;
   Dispose(Qd);
   Qd:=Pd;
  end;
end;
```

Metoda ***TPDrenBox.HandleEvent*** gestionează evenimentele asociate liniei de stare, prin apelul metodei ***Tview.HandleEvent***, apoi prin testul producerii a trei tipuri speciale de evenimente (mouse, key, broadcast).

```
Procedure TPDrenBox.HandleEvent(var Event: TEvent);
var
  R: TRect;
begin
  inherited HandleEvent(Event);
  if Event.What = evCommand then
  begin
    case Event.Command of
      cmRetine:
        Begin
          If CurrentOrder > MaxDren Then
           Begin
```

```
            DisableCommands([cmUrmator])
           end
          else
           Begin
            EnableCommands([cmAnterior]);
            Retine;
            If CurrentOrder = MaxDren Then
             Begin
              CurrentOrder:=1;
              Counter^.SetCurrent(CurrentOrder);
              Td:=Pd;
             end
            else
             Begin
              CurrentOrder:=CurrentOrder+1;
              Counter^.SetCurrent(CurrentOrder);
              Td:=Td^.urm;
             end;
            Show;Dispose(Drenaj);
            ClearEvent(Event);
           end;
        end;
      cmAnterior:
        Begin
          CurrentOrder:=CurrentOrder-1;
          Counter^.SetCurrent(CurrentOrder);
          Td:=Td^.ant;
          Show;Dispose(Drenaj);
          ClearEvent(Event);
        end;
      cmUrmator:
        Begin
          CurrentOrder:=CurrentOrder+1;
          Counter^.SetCurrent(CurrentOrder);
          Td:=Td^.urm;
          Show;Dispose(Drenaj);
          ClearEvent(Event);
        end;
    end;
  end;
end;
```

7.1.6 Submeniul; Puncte interioare.

Acest submeniu este folosit pentru editarea (introducerea, vizualizarea şi/sau modificarea) datelor punctele interioare din interiorul domeniului. Utilizatorului i se crează în acest sens facilitatea de a edita următoarele date:

- Coordonatele punctelor interioare (abscisă, ordonată, cotă);

Procedurile şi funcţiile folosite pentru editarea din cadrul acestui submeniu se afla în unitul ***Pinter.pas***. În cadrul acestui unit punctele interioare sunt definite ca fiind obiecte de tipul ***TPInt*** în care X, Y, Z sunt câmpurile sale.

Totodată obiectul ***TPInt*** mai conţine în plus şi elementele ***ant*** şi ***urm*** care sunt de tip referinţă la obiectul ***TPInt***. Şi în acest caz, deoarece mulţimea elementelor care alcătuiesc punctele interioare se află într-o structură de tip listă, elementele de tip referinţă ***ant*** şi ***urm*** sunt folosite pentru deplasarea şi poziţionarea pe un anumit element din cadrul listei.

Pentru a putea vizualiza câmpurile obiectului ***TPInt*** tot în cadrul unitului ***PInter.pas*** este definit obiectul ***TPIntBox***.

```
Type

  PPint = ^TPint;

  TPInt = Object
    X     : String[8];
    Y     : String[8];
    Z     : String[8];
    urm,ant   : PPint;
  end;



  PIntBox = ^TPIntBox;
   TPIntBox = Object(TDialog)
   X,Y,Z :PInputLine;
   Counter:PCountView;
   Constructor Init;
   Procedure HandleEvent(var Event: TEvent);virtual;
   Procedure Retine;
   Procedure Show;
  end;
```

Câmpurile obiectului ***TPIntBox*** sunt X, Y, Z de tip ***PinputLine***. Pentru introducerea valorilor acestor câmpuri se folosesc obiecte de tip ***TInputLine***, derivate din ***TView***, al cărui constructor crează o linie în care se pot înscrie un anumit număr de caractere. Şi în acest caz pentru etichetarea fiecărei linii cu un text explicativ se utilizează obiectul ***TLabel***, derivat din ***TStaticText***, al cărui constructor crează o etichetă, pusă în evidenţă pe ecranul monitorului în momentul selectării unuia din câmpuri. În continuare este exemplificat modul de realizare al ferestrei pentru editarea valorilor în cazul punctelor interioare.

```
R.Assign(0, 0, 60, 17);
inherited Init(R, 'Puncte in interiorul domeniului');
Options := Options or ofCentered;
HelpCtx := $F000;


R.Assign(16, 2, 28, 3);
X := New(PInputLine, Init(R, 8));
Insert(X);
R.Assign(4, 2, 14, 3);
Insert(New(PLabel, Init(R, '~X~ Punct:', X)));


R.Assign(16, 4, 28, 5);
Y := New(PInputLine, Init(R, 8));
Insert(Y);
R.Assign(4, 4, 14, 5);
Insert(New(PLabel, Init(R, '~Y~ Punct:', Y)));


R.Assign(16, 6, 28, 7);
Z := New(PInputLine, Init(R, 8));
Insert(Z);
R.Assign(4, 6, 14, 7);
Insert(New(PLabel, Init(R, '~Z~ Punct:', Z)));
```

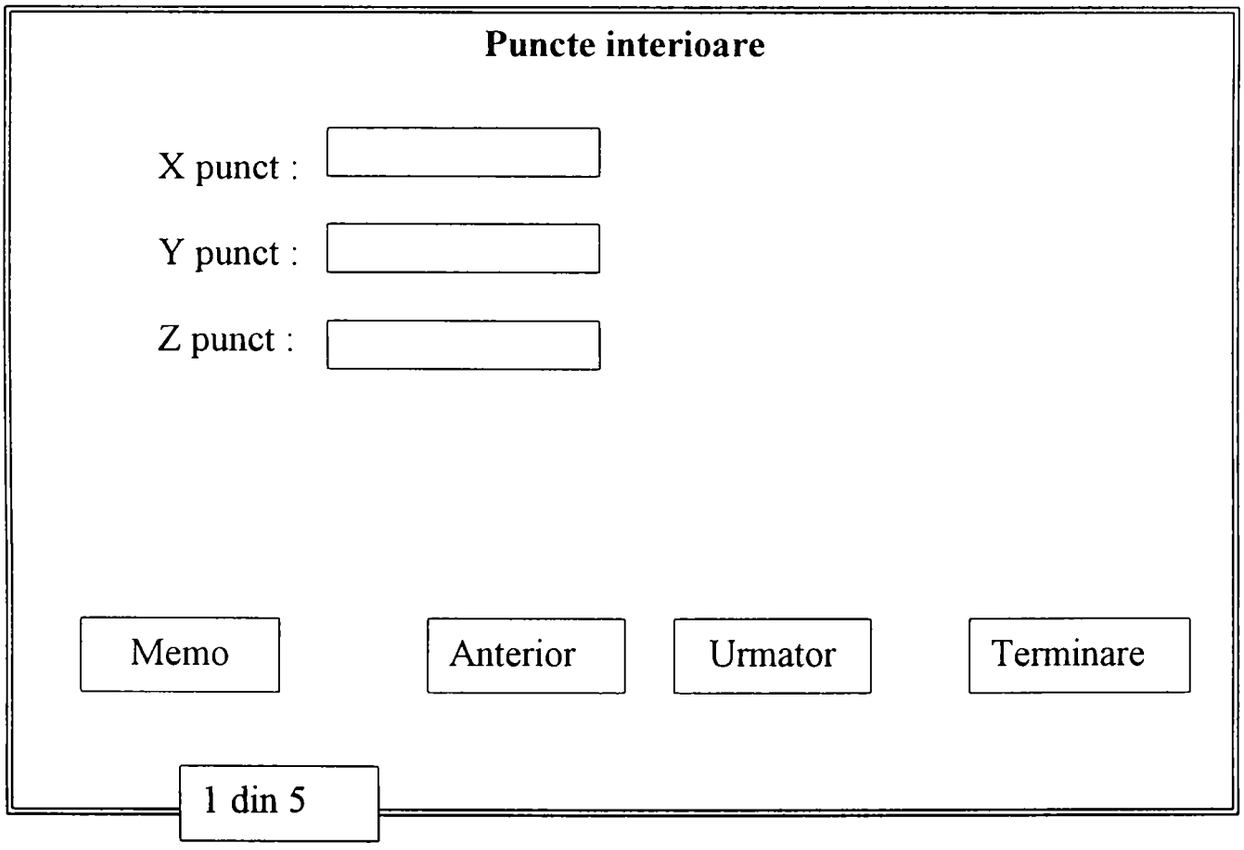


Fig. 7.7 Meniul de editare pentru puncte interioare

Constructorul ***TPIntBox.Init*** inițializează de fapt fereastra de dialog, care se poate deplasa şi închide dimensiunile ei rămânând nemodificabile. În această fereastră se fixează şi butoanele **Memo**, **Anterior**, **Următor** şi **Închide**, se inițializează valoarea maximă a contorului la o valoare egală cu MaxInt (numărul maxim de puncte interioare din cadrul domeniului), valoarea curentă fiind egală cu 1, după care folosind metoda ***Show*** se inițializează câmpurile obiectului ***TDrenaj*** cu valorile corespunzătoare numărului curent elementului şi le afişează în cadrul fereastrei.

```
Constructor TPIntBox.init;

var
  R: TRect;

begin

  R.Assign(0, 0, 60, 17);
  inherited Init(R, 'Puncte in interiorul domeniului');
  Options := Options or ofCentered;
  HelpCtx := $F000;


  R.Assign(16, 2, 28, 3);
  X := New(PInputLine, Init(R, 8));
  Insert(X);
  R.Assign(4, 2, 14, 3);
  Insert(New(PLabel, Init(R, '~X~ Punct:', X)));

  R.Assign(16, 4, 28, 5);
  Y := New(PInputLine, Init(R, 8));
  Insert(Y);
  R.Assign(4, 4, 14, 5);
  Insert(New(PLabel, Init(R, '~Y~ Punct:', Y)));

  R.Assign(16, 6, 28, 7);
  Z := New(PInputLine, Init(R, 8));
  Insert(Z);
  R.Assign(4, 6, 14, 7);
  Insert(New(PLabel, Init(R, '~Z~ Punct:', Z)));


  R.Assign(2, 14, 12, 16);
  Insert(New(PButton, Init(R, '~O~k', cmRetine, bfNormal)));
  R.Assign(20, 14, 32, 16);
  Insert(New(PButton, Init(R, '~A~nterior', cmAnterior, bfNormal)));
  R.Assign(31, 14, 43, 16);
  Insert(New(PButton, Init(R, '~U~rmator', cmUrmator, bfNormal)));
  R.Assign(46, 14, 57, 16);
  Insert(New(PButton, Init(R, '~A~nulare', cmCancel, bfNormal)));


  CurrentOrder := 1;
```

```
  R.Assign(5, 16, 20, 17);
  Counter := New(PCountView, Init(R));
  with Counter^ do
  begin
    SetCount(MaxInt);
    SetCurrent(CurrentOrder);
  end;

  Insert(Counter);
  SelectNext(False);

  DisableCommands([cmAnterior]);
  T:=P;Show;
end;
```

Metoda ***TPIntBox.Show*** , are funcția de a inițializa câmpurile obiectului ***TPInt*** cu valorile corespunzătoare elementului curent. Prin procedura ***SetData*** moştenită de la ***TView***, se copiază un număr de octeți din parametru în structura internă a obiectului, în cazul de față fereastra de dialog. În plus metoda activează sau dezactivează comenzile ***cmUrmător***, ***cmAnterior*** funcție de valoarea lui CurrentOrder care indică poziția elementului curent.

```
Procedure TPIntBox.Show;

Begin

   If CurrentOrder >= MaxInt Then
      DisableCommands([cmUrmator])
   else
      EnableCommands([cmUrmator]);
   If CurrentOrder <=1 Then
      DisableCommands([cmAnterior])
   else
      EnableCommands([cmAnterior]);
   New(PInt);
   With PInt^ Do
     Begin
      X:=T^.X;
      Y:=T^.Y;
      Z:=T^.Z;
     end;
   SetData(PInt^);
end;
```

Metoda ***TPIntBox.Retine*** are funcția de a prelua folosind metoda ***GetData*** datele care se află în fereastra de dialog şi de a inițializa câmpurile obiectului TBox cu aceste valori.

```
Procedure TPIntBox.Retine;

Begin

  GetData(PInt^);
  With T^ Do
    Begin
      X:=PInt^.X;
      Y:=PInt^.Y;
      Z:=PInt^.Z;
    end;
end;
```

Destructorul ***TPIntBox.Distruge***, realizează eliberarea zonei de memorie care a fost ocupată de către lista care a conținut punctele interioare ale domeniului. Variabilele dinamice referite se distrug prin apelul procedurii **Dispose**.

```
Destructor TPIntBox.Distruge;
Begin
 Q:=P;
 While P <> nil Do
  Begin
   P:=Q^.urm;
   Dispose(Q);
   Q:=P;
  end;
end;
```

Metoda ***TPIntBox.HandleEvent*** gestionează evenimentele asociate liniei de stare, prin apelul metodei ***TView.HandleEvent***, apoi prin testul producerii a trei tipuri speciale de evenimente (mouse, key, broadcast).

```
Procedure TPIntBox.HandleEvent(var Event: TEvent);

var
  R: TRect;

begin
  inherited HandleEvent(Event);
  if Event.What = evCommand then
  begin
    case Event.Command of
      cmRetine:
        Begin
          If CurrentOrder > MaxInt Then
           Begin
            DisableCommands([cmUrmator])
           end
          else
           Begin
            EnableCommands([cmAnterior]);
```

```
            Retine;
            If CurrentOrder = MaxInt Then
              Begin
               CurrentOrder:=1;
               Counter^.SetCurrent(CurrentOrder);
               T:=P;
              end
            else
              Begin
               CurrentOrder:=CurrentOrder+1;
               Counter^.SetCurrent(CurrentOrder);
               T:=T^.urm;
              end;
            Show;
            ClearEvent(Event);
           end;
        end;
      cmAnterior:
        Begin
          CurrentOrder:=CurrentOrder-1;
          Counter^.SetCurrent(CurrentOrder);
          T:=T^.ant;Show;
          ClearEvent(Event);
        end;
      cmUrmator:
        Begin
          CurrentOrder:=CurrentOrder+1;
          Counter^.SetCurrent(CurrentOrder);
          T:=T^.urm;Show;
          ClearEvent(Event);
        end;
  end;
 end;
end;
```

7.1.7 Submeniul; Puțuri imperfecte

Acest submeniu conține la rândul său încă două submeniuri. Primul submeniu ***Parametrii puțuri*** este folosit pentru editarea următoarelor date:

- Coordonatele puțurilor imperfecte (abscisă, ordonată);
- Numărul de elemente ale puțurilor (discretizarea lor);
- Grosimea stratului acvifer;
- Raza puțului imperfect.

Al doilea submeniu ***Parametrii elementelor*** este folosit pentru discretizarea puțului imperfect, respectiv pentru editarea cotei elementelor care alcătuiesc puțul imperfect precum şi a potențialului fiecărui element al puțului imperfect.

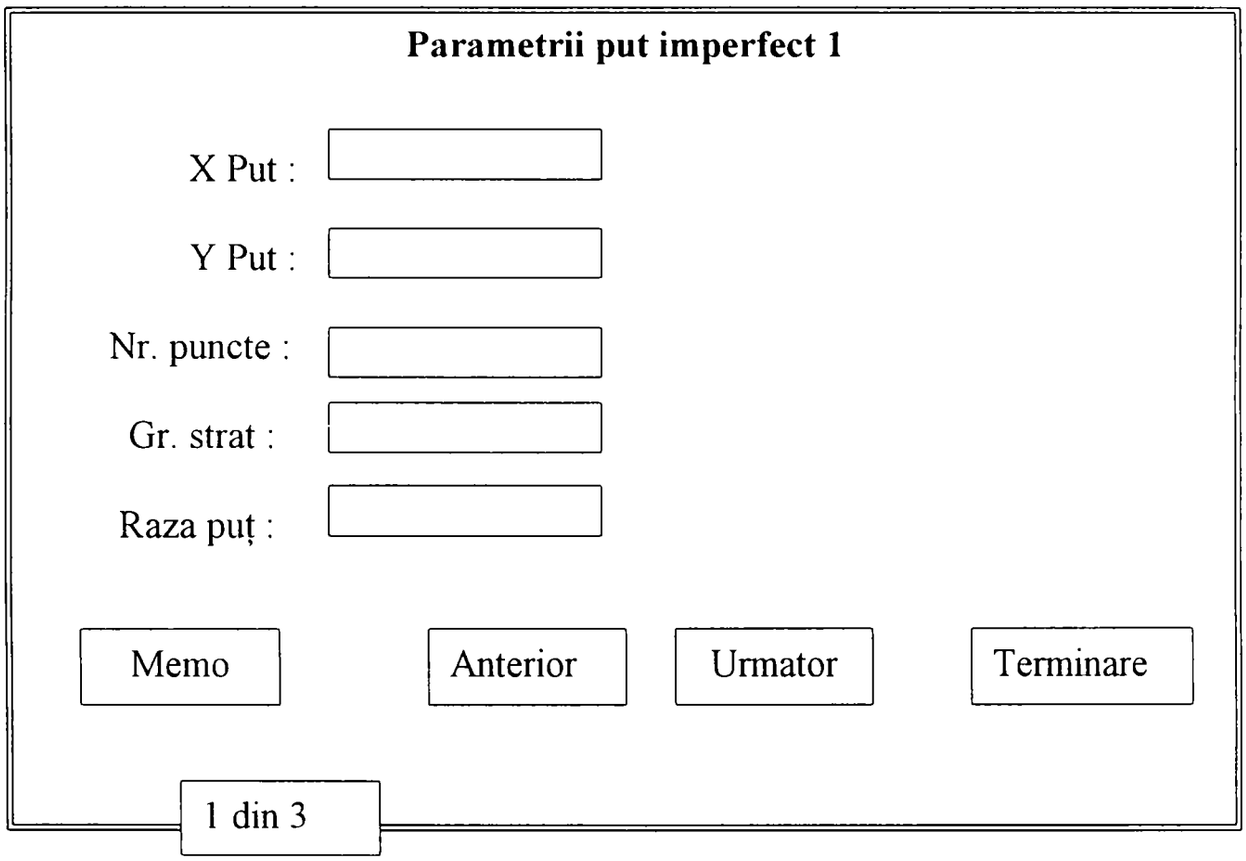


Fig. 7.8 Meniul de editare pentru parametrii puțurilor imperfecte

Procedurile şi funcțiile folosite pentru editarea din cadrul acestui submeniu se afla în unitul ***Putimper.pas***. În cadrul acestui unit puțurile imperfecte sunt definite ca fiind obiecte de tipul ***TPutimp*** în care X, Y, Puncte, T, Raza sunt câmpurile sale.

Şi în acest caz obiectul ***TPutimp*** mai conține în plus şi elementele ***ant*** şi ***urm*** care sunt de tip referință la obiectul ***TPutimp***. Deoarece mulțimea elementelor care alcătuiesc puțurile imperfecte se află într-o structură de tip listă, elementele de tip referință ***ant*** şi ***urm*** sunt folosite pentru deplasarea şi poziționarea pe un anumit puț imperfect din cadrul listei.

```
Type
  PPutImp = ^TPutImp;

  TPutImp = Object
    X      : String[8];
    Y      : String[8];
    Puncte : String[8];
    T      : String[8];
    Raza   : String[8];
    ant,urm: PPutImp;
  end;
```

Pentru a putea vizualiza câmpurile obiectului ***TPutimp*** tot în cadrul unitului ***Putimper.pas*** este definit obiectul ***TputImpBox***. Unitul ***Lambd.pas*** conține elementele necesare editării coeficientului *lambda* în cazul în care mişcarea în interiorul puțului imperfect este cu pierdere de sarcină.

```
  PPutImpBox = ^TPutImpBox;
   TPutImpBox = Object(TDialog)
   X,Y,Puncte,T,Raza:PInputLine;
   Counter:PCountView;
   Constructor Init;
   Procedure HandleEvent(var Event: TEvent);virtual;
   Procedure Show;
   Procedure Retine;
  end;
```

```
Unit Lambd;

Interface
Uses Objects,Dialogs,Drivers,Iteratii;

Type
  PLam = ^TLam;
  TLam = Object
    Lamb    : string[3];
  end;
  LambdaBox = ^CoefBox;
   CoefBox = Object(TDialog)
   Lamb:PInputLine;
   Constructor Init;
   Procedure HandleEvent(var Event: TEvent);virtual;
   Procedure Conversie;
  end;
```

Câmpurile obiectului ***TPutimpBox*** sunt X, Y, Puncte, T şi Raza de tip ***PinputLine***. Pentru introducerea valorilor acestor câmpuri se folosesc obiecte de tip ***TInputLine***, derivate din ***TView***, al cărui constructor crează o linie în care se pot înscrie un anumit număr de caractere. Şi în acest caz pentru etichetarea fiecărei linii cu un text explicativ se utilizează obiectul ***TLabel***, derivat din ***TStaticText***, al cărui constructor crează o etichetă, pusă în evidență pe ecranul monitorului în momentul selectării unuia din câmpuri. În continuare este exemplificat modul de realizare al ferestrei pentru editarea valorilor în cazul puțurilor imperfecte.

```
  R.Assign(0, 0, 60, 17);
  Str(j,Sir);
  inherited Init(R, 'Put imperfect '+Sir);
  Options := Options or ofCentered;
  HelpCtx := $F000;
```

```
R.Assign(16, 2, 28, 3);
X := New(PInputLine, Init(R, 8));
Insert(X);
R.Assign(4, 2, 14, 3);
Insert(New(PLabel, Init(R, '~X~ Put   :', X)));

R.Assign(16, 4, 28, 5);
Y := New(PInputLine, Init(R, 8));
Insert(Y);
R.Assign(4, 4, 14, 5);
Insert(New(PLabel, Init(R, '~Y~ Put   :', Y)));

R.Assign(16, 6, 28, 7);
Puncte := New(PInputLine, Init(R, 8));
Insert(Puncte);
R.Assign(4, 6, 14, 7);
Insert(New(PLabel, Init(R, '~N~r.pcte :', Puncte)));

R.Assign(16, 8, 28, 9);
T := New(PInputLine, Init(R, 8));
Insert(T);
R.Assign(4, 8, 14, 9);
Insert(New(PLabel, Init(R, '~G~r.Strat:', T)));


R.Assign(16, 10, 28, 11);
Raza := New(PInputLine, Init(R, 8));
Insert(Raza);
R.Assign(4, 10, 14, 11);
Insert(New(PLabel, Init(R, '~R~aza put:', Raza)));
```

Pentru obiectul ***TLam*** realizarea ferestrei pentru editarea valorilor are următoarea formă:

```
R.Assign(0, 0, 30, 7);
inherited Init(R, 'Coeficientul Lambda ');
Options := Options or ofCentered;
HelpCtx := $F000;

R.Assign(16, 2, 28, 3);
Lamb := New(PInputLine, Init(R, 8));
Insert(Lamb);
R.Assign(4, 2, 14, 3);
Insert(New(PLabel, Init(R, '~L~ambda  :', Lamb)));
```

Constructorul ***TPutImpBox.Init*** inițializează de fapt fereastra de dialog, care se poate deplasa şi închide dimensiunile ei rămânând nemodificabile. În această

fereastră se fixează şi butoanele **Memo**, **Anterior**, **Următor** şi **Închide**, se iniţializează valoarea maximă a contorului la o valoare egală cu MaxImper (numărul maxim de puţuri imperfecte din cadrul domeniului), valoarea curentă fiind egală cu 1, după care folosind metoda ***Show*** se iniţializează câmpurile obiectului ***TPutImp*** cu valorile corespunzătoare numărului curent puţului imperfect şi le afişează în cadrul fereastrei.

```
Constructor TPutImpBox.init;

var
  R   : TRect;
  Sir :String;
begin

  R.Assign(0, 0, 60, 17);
  Str(j,Sir);
  inherited Init(R, 'Put imperfect '+Sir);
  Options := Options or ofCentered;
  HelpCtx := $F000;


  R.Assign(16, 2, 28, 3);
  X := New(PInputLine, Init(R, 8));
  Insert(X);
  R.Assign(4, 2, 14, 3);
  Insert(New(PLabel, Init(R, '~X~ Put   :', X)));

  R.Assign(16, 4, 28, 5);
  Y := New(PInputLine, Init(R, 8));
  Insert(Y);
  R.Assign(4, 4, 14, 5);
  Insert(New(PLabel, Init(R, '~Y~ Put   :', Y)));

  R.Assign(16, 6, 28, 7);
  Puncte := New(PInputLine, Init(R, 8));
  Insert(Puncte);
  R.Assign(4, 6, 14, 7);
  Insert(New(PLabel, Init(R, '~N~r.pcte :', Puncte)));

  R.Assign(16, 8, 28, 9);
  T := New(PInputLine, Init(R, 8));
  Insert(T);
  R.Assign(4, 8, 14, 9);
  Insert(New(PLabel, Init(R, '~G~r.Strat:', T)));


  R.Assign(16, 10, 28, 11);
  Raza := New(PInputLine, Init(R, 8));
  Insert(Raza);
  R.Assign(4, 10, 14, 11);
  Insert(New(PLabel, Init(R, '~R~aza put:', Raza)));
```

```
  R.Assign(2, 14, 12, 16);
  Insert(New(PButton, Init(R, '~O~k', cmRetine, bfNormal)));
  R.Assign(24, 14, 34, 16);
  Insert(New(PButton, Init(R, '~A~nter', cmAnterior, bfNormal)));
  R.Assign(35, 14, 45, 16);
  Insert(New(PButton, Init(R, '~U~rmat', cmUrmator, bfNormal)));
  R.Assign(46, 14, 56, 16);
  Insert(New(PButton, Init(R, '~A~nulare', cmCancel, bfNormal)));

  CurrentOrder := 1;

  R.Assign(5, 16, 20, 17);
  Counter := New(PCountView, Init(R));
  with Counter^ do
  begin
    SetCount(MaxImper);
    SetCurrent(CurrentOrder);
  end;

  Insert(Counter);
  SelectNext(False);
  DisableCommands([cmAnterior]);
  Ti:=Pi;
  Show;
end;
```

În cazul coeficientului lambda constructorul are următoarea formă:

```
Constructor CoefBox.init;

var
  R    : TRect;

Begin
  R.Assign(0, 0, 30, 7);
  inherited Init(R, 'Coeficientul Lambda ');
  Options := Options or ofCentered;
  HelpCtx := $F000;

  R.Assign(16, 2, 28, 3);
  Lamb := New(PInputLine, Init(R, 8));
  Insert(Lamb);
  R.Assign(4, 2, 14, 3);
  Insert(New(PLabel, Init(R, '~L~ambda  :', Lamb)));

  R.Assign(3, 4, 13, 6);
  Insert(New(PButton, Init(R, '~O~k', cmRetineLambda, bfNormal)));


  R.Assign(15, 4, 25, 6);
  Insert(New(PButton, Init(R, '~C~lose', cmOk, bfNormal)));
  SelectNext(False);
end;
```

Metoda ***TPutimpBox.Show*** , are funcția de a inițializa câmpurile obiectului ***TPutimp*** cu valorile corespunzătoare elementului curent. Prin procedura ***SetData*** moştenită de la ***TView***, se copiază un număr de octeți din parametru în structura internă a obiectului, în cazul de față fereastra de dialog. În plus metoda activează sau dezactivează comenzile ***cmUrmător***, ***cmAnterior*** funcție de valoarea lui CurrentOrder care indică poziția elementului curent.

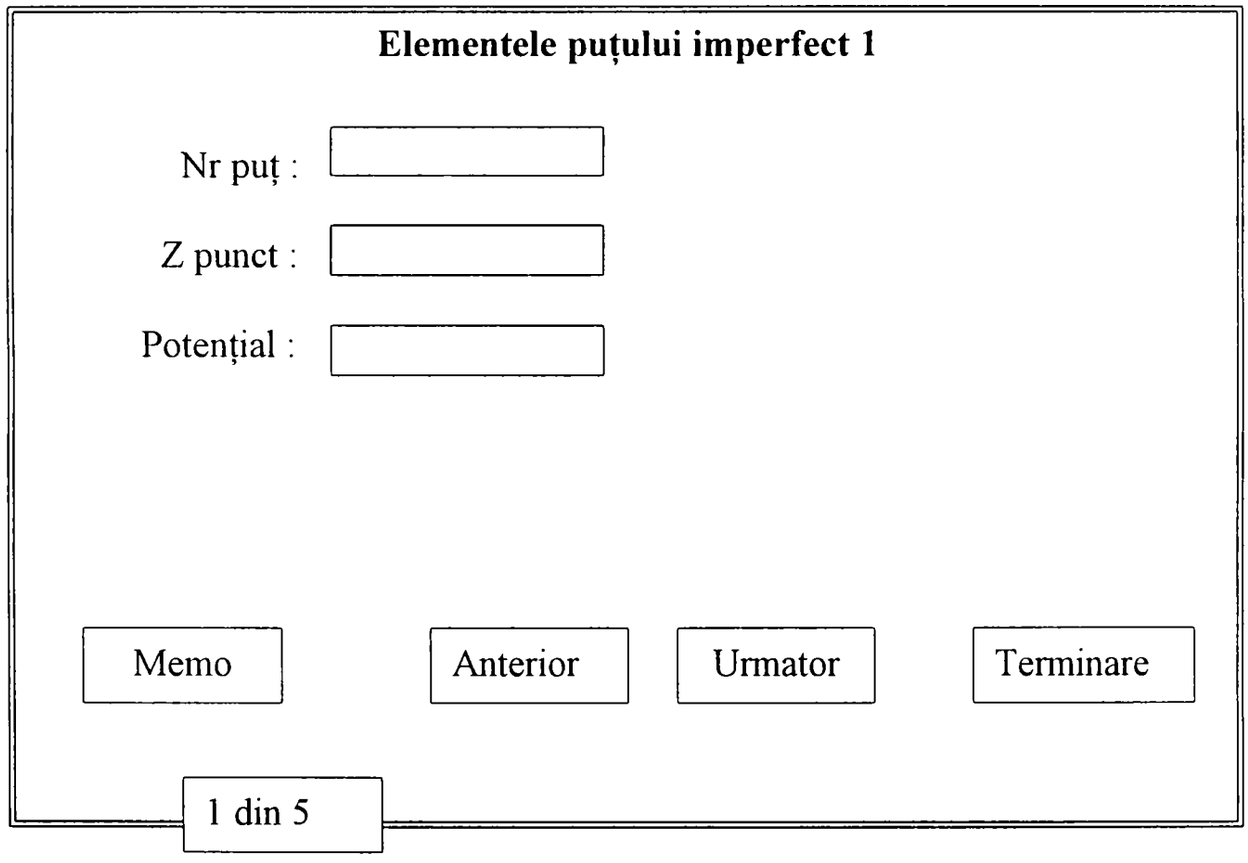


Fig. 7.9 Meniul de editare al discretizării puțurilor imperfecte

```
Procedure TPutImpBox.Show;

Begin


   If CurrentOrder >= MaxImper Then
      DisableCommands([cmUrmator])
   else
      EnableCommands([cmUrmator]);

   If CurrentOrder <=1 Then
      DisableCommands([cmAnterior])
   else
```

```
      EnableCommands([cmAnterior]);
   New(PutImp);
   With PutImp^ Do
     Begin
      X:=Ti^.x;
      Y:=Ti^.y;
      Puncte:=Ti^.Puncte;
      T:=Ti^.T;
      Raza:=Ti^.Raza;
     end;
   SetData(PutImp^);
end;
```

Metoda ***TPutImpBox.Retine*** are funcția de a prelua folosind metoda ***GetData*** datele care se află în fereastra de dialog şi de a iniția câmpurile obiectului TBox cu aceste valori.

```
Procedure TPutImpBox.Retine;

Var
   code:Integer;

Begin
  GetData(PutImp^);
  With Ti^ Do
   Begin
     X:=PutImp^.X;
     Y:=PutImp^.Y;
     Puncte:=PutImp^.Puncte;
     T:=PutImp^.T;
     Raza:=PutImp^.Raza;
   end;
  Val(Ti^.Puncte,ElImp[CurrentOrder],code);
end;
```

Destructorul ***TPutImpBox.Distruge***, realizează eliberarea zonei de memorie care a fost ocupată de către lista care a conținut puțurile imperfecte ale domeniului. Variabilele dinamice referite se distrug prin apelul procedurii **Dispose**.

```
Destructor TPutImpBox.Distruge;

Begin
 Qi:=Pi;
 While Pi <> nil Do
  Begin
```

```
    Pi:=Qi^.urm;
    Dispose(Qi);
    Qi:=Pi;
  end;
end;
```

Metoda ***TPutImpBox.HandleEvent*** gestionează evenimentele asociate liniei de stare, prin apelul metodei ***TView.HandleEvent***, apoi prin testul producerii a trei tipuri speciale de evenimente (mouse, key, broadcast).

```
Procedure TPutImpBox.HandleEvent(var Event: TEvent);
var
  R: TRect;
begin
  inherited HandleEvent(Event);
  if Event.What = evCommand then
  begin
    case Event.Command of
      cmRetine:
       Begin
        If CurrentOrder > MaxImper Then
         Begin
          DisableCommands([cmUrmator])
         end
        else
         Begin
          EnableCommands([cmAnterior]);
          Retine;
          If CurrentOrder = MaxImper Then
            Begin
             CurrentOrder:=1;
             Counter^.SetCurrent(CurrentOrder);
             Ti:=Pi;
            end
          else
            Begin
             CurrentOrder:=CurrentOrder+1;
             Counter^.SetCurrent(CurrentOrder);
             Ti:=Ti^.urm;
            end;
          Show;{Dispose(PutImp)};
          ClearEvent(Event);
         end;
        end;
      cmAnterior:
        Begin
          CurrentOrder:=CurrentOrder-1;
          Counter^.SetCurrent(CurrentOrder);
          Ti:=Ti^.ant;
          Show;{Dispose(PutImp);}
          ClearEvent(Event);
```

```
        end;
      cmUrmator:
        Begin
          CurrentOrder:=CurrentOrder+1;
          Counter^.SetCurrent(CurrentOrder);
          Ti:=Ti^.urm;
          Show;{Dispose(PutImp);}
          ClearEvent(Event);
        end;
  end;
 end;
end;
```

În cazul unitului ***Lambd.pas***

```
procedure CoefBox.HandleEvent(var Event: TEvent);
var
  R: TRect;

Begin
  inherited HandleEvent(Event);
  if Event.What = evCommand then
  begin
    case Event.Command of
      cmRetineLambda:
       Begin
        Conversie;
        ClearEvent(Event);
       end;
    end;
  end;
end;
```

7.1.8 Submeniul; Subdomenii

Acest submeniu conține la rândul său încă două submeniuri. Primul submeniu ***Parametrii subdomeniu*** este folosit pentru editarea următoarelor date:

- Numărul de ordine al subdomeniului;
- Numărul de elemente al subdomeniului;
- Aportul de precipitații pentru subdomeniul în cauză.

Al doilea submeniu ***Coordonate subdomeniu*** este folosit pentru editarea frontierei subdomeniilor acesta permițând editarea următoarelor date:

- Numărul punctului de pe frontieră;
- Abscisa punctului respectiv;
- Ordonata punctului respectiv.

Schematic cele două submeniuri se prezintă în felul următor:

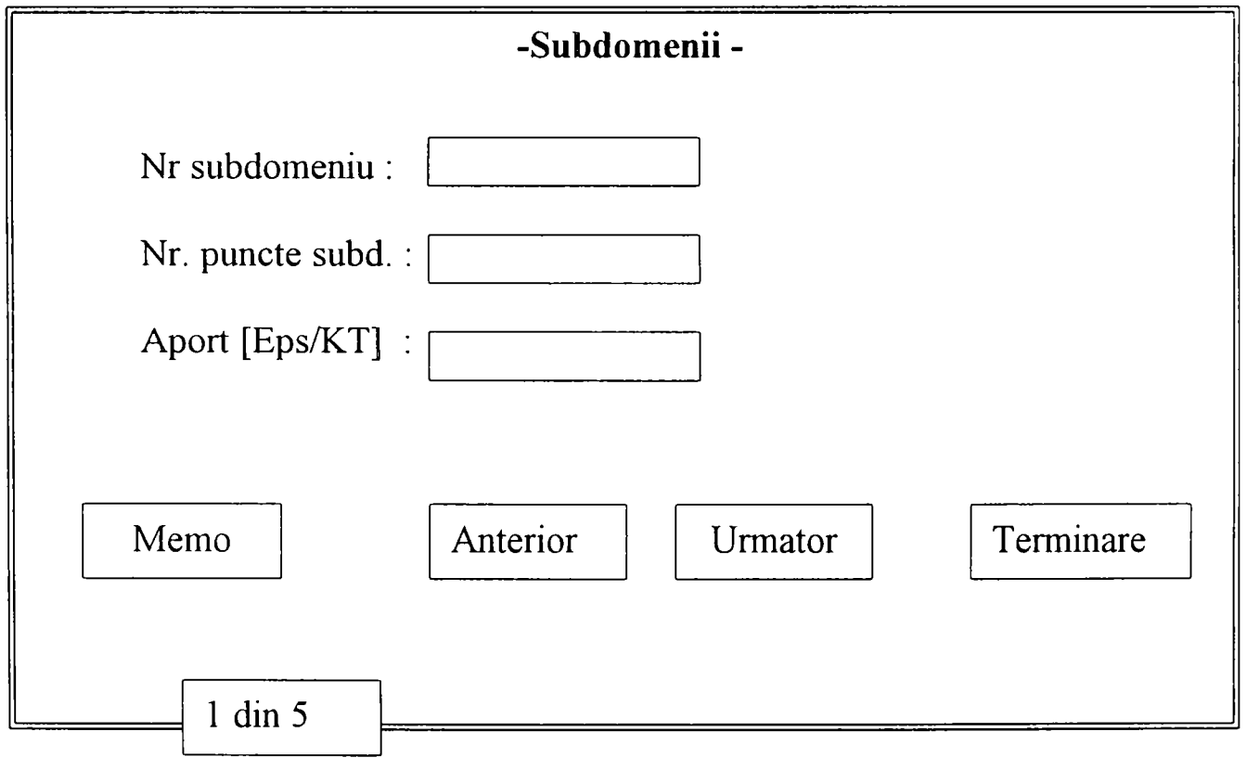


Fig. 7.10 Meniul de editare pentru subdomenii

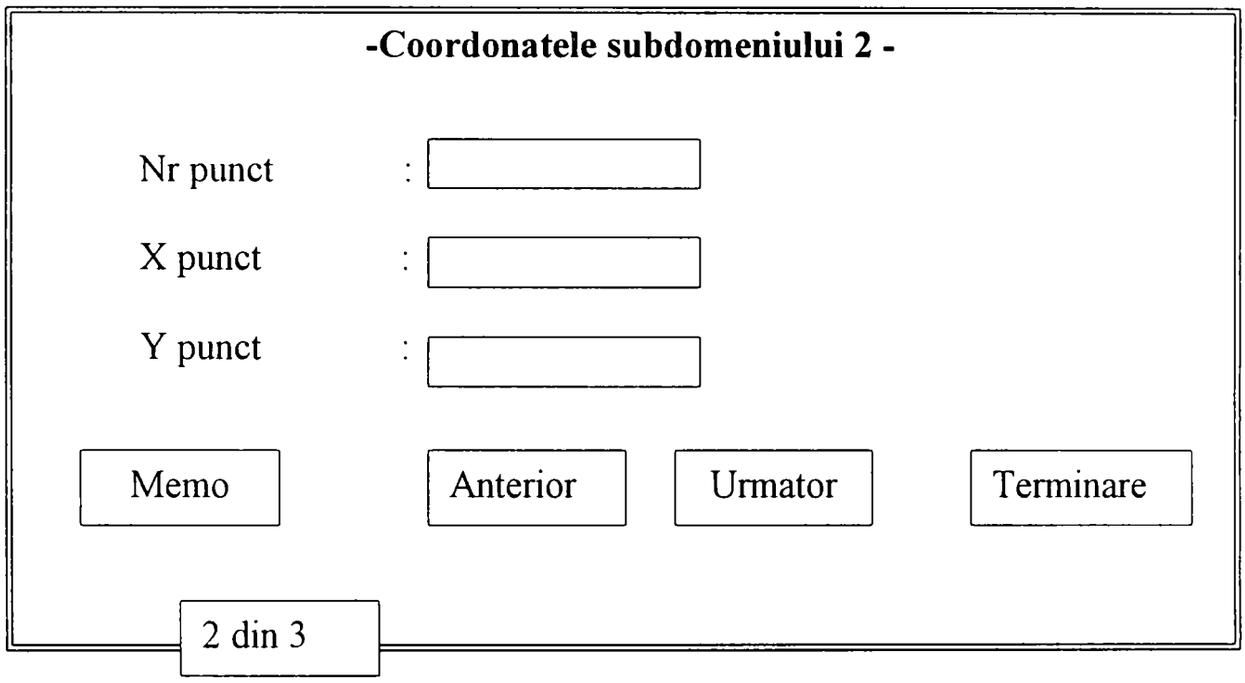


Fig. 7.11 Meniul de editare al discretizării subdomeniilor

Deoarece variabilele care intervin în cazul subdomeniilor este relativ mic nu s-a mai recurs la structuri de date dinamice ele fiind declarate ca variabile de tip tablou.

```
Type

     Coordonate_Subdomen = Object
       X,Y:Real;
     end;

Var
     Sub :Array[1..Max] of Coordonate_Subdomen;
```

7.1.9 Submeniul folosit pentru salvarea şi încărcarea datelor

În vederea rulării programului MEFRO setul de date introduse, respectiv editate prin intermediul submeniurilor prezentate anterior trebuie să fie salvat sub forma unui fişier de date. În acest sens se va putea crea pentru fiecare aplicaţie în parte un fişier de date caracteristic aplicaţiei. Pentru realizarea submeniului de salvare al datelor s-a folosit ferestra de dialog standard ***TFileDialog***, descendentă din ***TDialog***.

TfileDialog asigură interfaţa necesară pentru alegerea unui nume de fişier din catalogul dorit sau introducerea unuia nou. Această interfaţă este similară celei folosite pentru alegerea unui fişier chiar de către Borland Pascal. ***TfileDialog*** asigură doar posibilitatea de a selecta comod un fişier, prelucrarea efectivă a fişierului se va realiza prin intermediul procedurilor ***Salvează*** şi ***Citeşte***.

```
If ExecuteDialog(New(PFileDialog,Init('*.fro', 'Salveaza fisier cu date',
  '~N~ume fisier', fdOkButton, 100)), @FileName) <> cmCancel then
   Salveaza;
```

Salvează reprezintă procedura de salvare a setului de date, numele fişierului de date trebuind să conţină maxim 8 caractere alfanumerice, extensia sau specificatorul de fişier fiind implicit ***.fro***.

Pentru încărcarea unui set de date se foloseşte tot fereastra standard de dialog standard ***TFileDialog***, descendentă din ***TDialog***. Proceduta ***Citeşte*** este folosită pentru deschiderea fişierului de date al aplicaţiei şi citirea datelor sale. Implicit specificatorul de fişier este ***.fro***.

```
FileName := '*.fro';
If ExecuteDialog(New(PFileDialog, Init('*.fro', 'Incarca fisierul cu
date','~N~ume fisier', fdOpenButton, 100)), @FileName) <> cmCancel then
       Citeste;
```

În ambele cazuri ferastra de dialog are o secţiune pentru fişiere, o listă a istoriei fişierelor folosite şi butoanele standard **Ok** şi **Cancel** folosite pentru validarea respectiv anularea comenzii de salvare sau încărcare.

7.2 Meniul Rulare program folosit în cadrul programului MEFRO.

În principiu prin apelarea acestui meniu, utilizatorului i se crează posibilitatea alegerii unui fişier de date (fişier care a fost creat anterior folosind meniul Editare date program împreună cu submeniurile asociate), care va fi citit flolosindu-se procedura ***Read_Data***. După citirea datelor, pasul următor constă în calculul şi asamblarea matricii Aji, care se efectuează prin apelul procedurii ***Calc_Aji***. Procedura ***Calc_Bji***, calculează termenii liberi ai sistemului de ecuații, respectiv asamblează matricea Bji. Odată construite aceste matrici procedura ***Asamblare*** crează matricea extinsă a sistemului de ecuații, matrice care este scrisă şi stocată sub forma unui fisier de date pe hardisk-ul calculatorului urmând ca procedura ***Solve_Sistem*** să realizeze rezolvarea acestui sistem de ecuații.

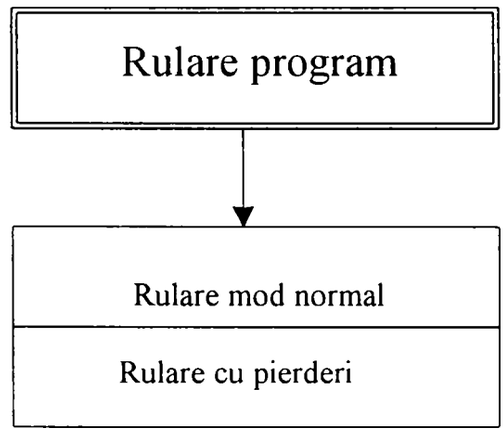


Fig. 7.12 Meniul Rulare al programului MEFRO

În continuare folosind rezultatele obținute în urma rezolvării sistemului de ecuații se obțin următoarele rezultate finale:

- potențialul în punctele mijlocii ale frontierei, procedura ***Pot_punct_mijlocii***;
- potențialul în puncte interioare ale domeniului, procedura ***Pot_puncte_interioare***;
- fluxul pe conturul domeniului, procedura ***Flux_pe_Contur***;
- valorile pierderilorde sarcină precum şi fluxul puțurilor imperfecte după caz folosind procedura ***Calcul_Pierderi***.

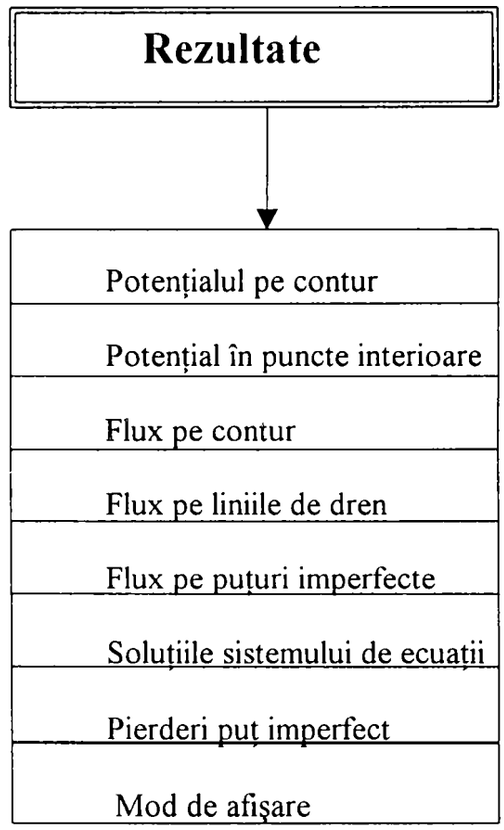


Fig. 7.13 Meniul Rezultate al programului MEFRO

Procedura ***Write_in_file*** este folosită pentru scrierea acestor rezultate finale în fişiere cu format ASCII, care au denumiri speciale funcție de conținutul informației pe care îl dețin.

Metoda apelată pentru execuția ferestrei de dialog în vederea citirii fişierului de date pentru a executa rularea programului este **CiteşteRun** şi are următoarea formă:

```
Procedure TEditorApp.CitesteRun(WildCard: PathStr);

Var
  FileName: FNameStr;


Begin
  FileName := '*.fro';
  if ExecuteDialog(New(PFileDialog, Init(WildCard, 'Incarca un fisier de
date',
  '~N~umele fisierului', fdOkButton, 100)), @FileName) <> cmCancel
  Then
   Begin
DisableCommands([cmRezContur,cmRezInt,cmRezFlux,cmRezPierderi,CmRezDren,c
mRezSol,cmFluxImper]);
    Assign(read_file,FileName);Mefrorez;
    EnableCommands([cmRezContur,cmRezSol,cmRezFlux]);
    If n1 > 0 Then EnableCommands([cmRezInt]);
    If n3 > 0 Then EnableCommands([cmRezDren]);
    If n7 > 0 Then EnableCommands([cmFluxImper]);
    If Pierderi Then EnableCommands([cmRezPierderi]);
   end;
end;
```

Se observă că după alegerea tipului de fişier se execută procedura ***Mefrorez*** al cărui corp este prezentat mai jos şi care care conţine procedurile menţionate şi descrise anterior

```
Procedure Mefrorez;


Begin
    ElibHeap;
    If Pierderi Then Iteratia:=Iteratia+1;
    Reset(read_file);
    Read_data;
    Close(read_file);
    Calc_Aji;
    Calc_Bji;
    Asamblare;
    Solve_sistem(Lin,rez);
    Pot_punct_mijlocii;
    Pot_puncte_interioare;
    Flux_pe_contur;
    Write_in_file;
    If Pierderi Then Calcul_pierderi;
    EndRun;
end;
```

7.2.1 Particularități specifice folosite în cadrul meniului Rulare.

Prima etapă după apelarea procedurii Mefrorez o reprezintă apelul procedurii ***ElibHeap***. Această procedură are rolul de a elibera zona Heap de memorie de eventualele variabile dinamice în vederea alocării unui spațiu cât mai mare pentru operațiile ulterioare. Procedura ElibHeap are următoarea formă:

```
Procedure ElibHeap;

Begin
    FerContur.Distruge;
    FerDren.Distruge;
    FerPut.Distruge;
    FerElemImp.Distruge;
    FerPutH.Distruge;
    FerPutQ.Distruge;
    FerInter.Distruge;
    FerSubDom.Distruge;
    FerSubCoor.Distruge;
    SingH.Sterge;
    SingQ.Sterge;
    Interior.Sterge;
    Boun.Sterge;
    LinDren.Sterge;
end;
```

ea aplefând destructorii specifici definiți în cadrul obiectelor existente.

Variabila *Pierderi* reprezintă un selector care are valoarea True în cazul în care se doreşte rezolvarea problemelor unde intervin puțuri imperfecte care prezintă în interior mişcare cu pierderi de sarcină (puțuri înisipate). Pentru restul tipurilor de probleme valoarea acestui selector este False.

În continuare urmează citirea datelor din finşierul de date selectat prin intermediul procedurii ***Read_Data***, care se află în unitul ***Sistem.pas*** calculul matricii sistemului de ecuații, folosind procedura ***Calc_Aji***, calculul termenilor liberi ai sistemului de ecuații, folosid procedura ***Calc_Bji*** şi calculul matricii extinse a sistemului de ecuații folosind matricile Aji şi Bji create prin intermediul procedurilor anterior menționate, prin procedura ***Asamblare*** care crează şi salvează matricea extinsă a sistemului de ecuații sub forma unui fişier ***aji.sis*** pe hardisk-ul unității de calcul.

7.2.2 Procedura numerică de rezolvare a sistemului de ecuații.

Rezolvarea sistemelor de ecuații algebrice este unul din capitolele analizei numerice frecvent abordat şi în cadrul căreia s-a făcut progrese substanțiale odată cu apariția calculatoarelor dotate cu procesoare prevăzute cu viteză mare lucru.

În general metodele de rezolvare a unui sistem de ecuații se grupează în două categorii: metode directe, bazate pe procedee de eliminare, şi metode iterative.

Pentru rezolvarea numerică a sistemelor de ecuații algebrice liniareexistă în literatură un număr mare de metode, unele din acestea fiind însă inutilizabile din punct de vedere al calculatorului. La alegerea unei metode de calcul, pentru rezolvarea unei anumite probleme pe un sistem de calcul dat, trebuie avute în vedere o serie de criterii cum ar fi: numărul secvențelor de calcul, precizia rezultatelor, posibilitatea introducerii unor teste de precizie pe parcursul desfăşurării algoritmului.

În cadrul metodelor directe, care se consideră a fi metode exacte, s-au impus metodele care au la bază algoritmul de eliminare Gauss, cu variantele Gauss-Jordan, Doolittle, Court, Cholesky [20], [36] , [46], [47], [58].

Metodele exacte permit obținerea soluției exacte a sistemului de ecuații, făcând abstracție de erorile de tip round-of ale sistemului de calcul, de erorile de rotunjire şi trunchiere, folosind un număr finit de operații elementare.

În cadrul metodelor iterative s-au impus metodele Jacobi, Gauss-Seidel, metoda relaxărilor succesive [37], [36], [55], metoda gradientului conjugat [50] şi metoda Lanczos [44].

Metodele iterative se caracterizează prin faptul că soluția sistemului considerat se obține ca limită a unui şir de valori ce reprezintă soluții pentru diverse iterații succesive. În cadrul acestor metode, se pune problema de a alege aceea dintre metode, care asigură cea mai mare viteză de convergență a soluțiilor pentru o aproximare inițială adecvat aleasă.

În ceea ce priveşte sistemul de ecuații care se formează în cadrul MEFRO, el este un sistem compatibil determinat, numarul de necunoscute fiind egal cu numarul de ecuații ale sistemului.

Metoda folosită la rezolvarea sistemului de ecuații este o metodă exactă, metoda Gauss (eliminare parțială) în care matricea A_{ji} a sistemului este triunghiularizată superior.

Opțiunea pentru această metodă este argumentată de faptul că, în rezolvarea unor sisteme de ecuații de mari dimensiuni, timpul de execuție pe calculator este proporțional cu numărul înmulțirilor simple ce trebuie efectuate pentru determinarea necunoscutelor. Din acest punct de vedere [6], [8], [7], [14], [50], metoda de eliminare directă Gauss comportă cel mai mic număr de operații. Pentru un sistem liniar cu n ecuații, numărul N de operații aritmetice necesare pentru eliminarea Gauss este [36]:

$$N = \frac{1}{6}\left(4n^3 + 9n^2 - 7n\right)$$

Transformările efectuate asupra matricei A pot fi rezumate la etapa k în formulele [58]:

$$a_{ij}^{(k)} = \begin{cases} a_{ij}^{(k-1)} \text{ pentru } i \le k-1 \text{ si oricare ar fi } j \in \overline{1,n} \\ 0 \quad \text{pentru } i \ge k,\ j \le k-1, \\ a_{ij}^{(k-1)} - \dfrac{a_{i,k-1}^{(k-1)}}{a_{k-1,k-1}^{(k-1)}} a_{(k-1),j}^{k-1)} \quad \text{pentru } i \ge k,\ j \ge k, \end{cases}$$

$$b_i^{(k)} = \begin{cases} b_i^{(k-1)} \quad \text{pentru } i \le k-1, \\ b_i^{(k-1)} - \dfrac{a_{i,k-1}^{(k-1)}}{a_{k-1,k-1}^{(k-1)}} b_{k-1}^{(k-1)} \quad \text{pentru } i \ge k \end{cases}$$

Soluția sistemului de ecuații va fi:

$$x_n = \frac{b_n^{(n)}}{a_{n,n}^{(n)}}$$

$$x_i = \frac{b_i^{(i)} - \sum_{j=i+1}^{n} a_{ij}^{(i)} x_j}{a_{ii}^{(n)}}, \quad i = n-1,\ n-2,\ \ldots,1.$$

Rezolvarea sistemului de ecuații se face apelând procedura ***Solve_Sistem*** care în baza metodei Gauss anterior prezentată, foloseşte matricea extinsă a sistemului creată şi salvată cu ajutorul procedurii ***Asamblare***.

Procedura are următorul conținut:

```
Procedure Solve_sistem(var ns:Integer;Var rez:Rezultat);


Var
   n,i,j,k:Longint;
   aii,aji,ajk,aik,anp,ann,an1:Real;
   t1,t2,t3,t4,term1,term2,tp1:Real;
   Fisier:File of real;



Begin
    Assign(fisier,'aji.dat');
    Reset(fisier);
    n:=ns;
    For i:=1 to n-1 Do
     Begin
      Seek(fisier,(i-1)*(n+1)+i-1);
      Read(fisier,aii);
        For j:=i+1 to n Do
          Begin
            Seek(fisier,(j-1)*(n+1)+i-1);
            Read(fisier,aji);
              For k:=n+1 DownTo i+1 Do
               Begin
                Seek(fisier,(j-1)*(n+1)+k-1);
                Read(fisier,ajk);
                Seek(fisier,(i-1)*(n+1)+k-1);
                Read(fisier,aik);
                ajk:=ajk-(aik/aii)*aji;
                Seek(fisier,(j-1)*(n+1)+k-1);
                Write(fisier,ajk);
               end;
          end;
     end;

     Seek(fisier,(n-1)*(n+1)+n);
     Read(fisier,anp);
     Seek(fisier,(n-1)*(n+1)+n-1);
     Read(fisier,ann);
     an1:=anp/ann;
     Seek(fisier,(n-1)*(n+1));
     Write(fisier,an1);
     For i:=n downto 2 Do
       Begin
         Seek(fisier,(i-2)*(n+1)+n);
         Read(fisier,t1);
         Seek(fisier,(i-1)*(n+1));
         Read(fisier,t2);
         Seek(fisier,(i-2)*(n+1)+i-1);
         Read(fisier,t3);
         Seek(fisier,(i-2)*(n+1)+i-2);
         Read(fisier,t4);
         tp1:=(t1-t2*t3)/t4;
         Seek(fisier,(i-2)*(n+1));
         Write(fisier,tp1);
```

```
          For j:=i-2 downto 1 Do
            Begin
             Seek(fisier,(j-1)*(n+1)+n);
             Read(fisier,term1);
             Seek(fisier,(j-1)*(n+1)+i-1);
             Read(fisier,term2);
             term1:=term1-t2*term2;
             Seek(fisier,(j-1)*(n+1)+n);
             Write(fisier,term1);
            end;
       end;

       For i:=1 to n Do
        Begin
          Seek(fisier,(i-1)*(n+1));
          Read(fisier,rez[i]);
        end;
       Close(fisier);
end;
```

Procedura lucrează deci cu un singur fişier **aji.sis** care conţine matricea extinsă a sistemului de ecuaţii şi care se autodistruge pe parcursul rezolvării sistemului. Datorită acestui mod de rezolvare se elimină stocarea matricii extinse a sistemului într-un tablou cu două dimensiuni ceea ce ar necesita ocuparea unei zone de memorie semnificativă ca şi mărime şi care în cazul unui volum mare de date ar putea conduce la imposibilitatea rulării programului. În acest mod numărul de ecuaţii poate fi considerabil mărit fără a mai apare probleme deosebite în ceea ce priveşte ocuparea memoriei calculatorului datorită faptului că pentru rezolvarea sistemului se folosesc fişiere stocate pe hardisk.

În urma rezolvării sistemului de ecuaţii se obţin soluţiile sistemului care reprezintă valori ale surselor de distribuţie şi care ulterior introduse în relaţiile de calcul ale înălţimii piezometrice şi ale fluxului pe contur permit determinarea acestor valori.

Rezultatele rulării programului sunt scrise în fişiere care pot fi vizualizate accesând meniul **Rezultate** din cadrul programului general MEFRO. Totodată, funcţie de necesităţi aceste rezultate pot fi tipărite la imprimantă.

7.3 Meniul Rezultate folosit în cadrul programului MEFRO.

În urma rulării unei aplicații MEFRO rezultatele rulării programului se vor afla în fișiere de tip text, cu conținut în format ASCII. Aceste fișiere au denumiri funcție de tipul datelor pe care le conțin. Astfel:

- **'Pot_pct.dat'** – conține potențialul calculat în punctele mijlocii ale elementelor de frontieră
- **'Pot_int.dat'** – conține potențialul calculat în punctele interioare ale domeniului
- **'Flc_con.dat'** – conține fluxul calculat pe frontiera domeniului
- **'Puturi.dat'** - conține fluxul calculat în cazul puțurilor imperfecte, pe fiecare element al puțului imperfect
- **'Flux_dre.dat'** - conține valori calculate ale fluxului de-a lungul liniilor de drenaj
- **'Sol_sis.dat'** – conține soluțiile sistemului de ecuații
- **'Imperrez.dat'** – conține valori calculate ale fluxului precum şi ale pierderilor de sarcină în cazul în care puțurile imperfecte prezintă mişcarea interioară cu pierderi de sarcină, succesiv pentru fiecare iterație.

Pentru exploatarea conținutului acestor fişiere s-a apelat la funcția OpenViz de tip TEditWindow. Conținutul acestei funcții este prezentat în cele ce urmează.

```
Function Aplicatia.OpenViz(FileName: FNameStr): PEditWindow;
var
  P: PView;
  R: TRect;
begin
  DeskTop^.GetExtent(R);
  P:=Application^.ValidView(New(PTextwindow,Init(R,Filename)));
  DeskTop^.Insert(P);
end;
```

Prin intermediul acestei funcții, corespunzător submeniului ales din cadrul meniului rezultate se realizează vizualizarea fişierului de date corespunzător submeniului selectat.

Conținutul fişierelor (rezultatele rulării MEFRO) pot fi prezentate simultan sub mai multe forme, de exemplu cărămidă sau cascadă. Folosind tastele definite cu acces direct, se poate activa următoarea fereastră, tasta F4, se poate închide fereastra curentă F3, sau se poate schimba numărul de linii afişate pe ecran F5. Aceste comenzi se află situate în linia de stare (Status line)

7.4 Meniul Listare date program folosit în cadrul programului MEFRO.

Acest meniu a fost creat pentru a putea oferi utilizatorului posibilitatea ca în urma unei aplicații să poată tipări la imprimantă rezultatele rulării aplicației şi/sau a datelor de intrare folosite pentru rularea aplicației.

Alcătuirea meniului împreună cu submeniurile este prezentată în cele ce urmează:

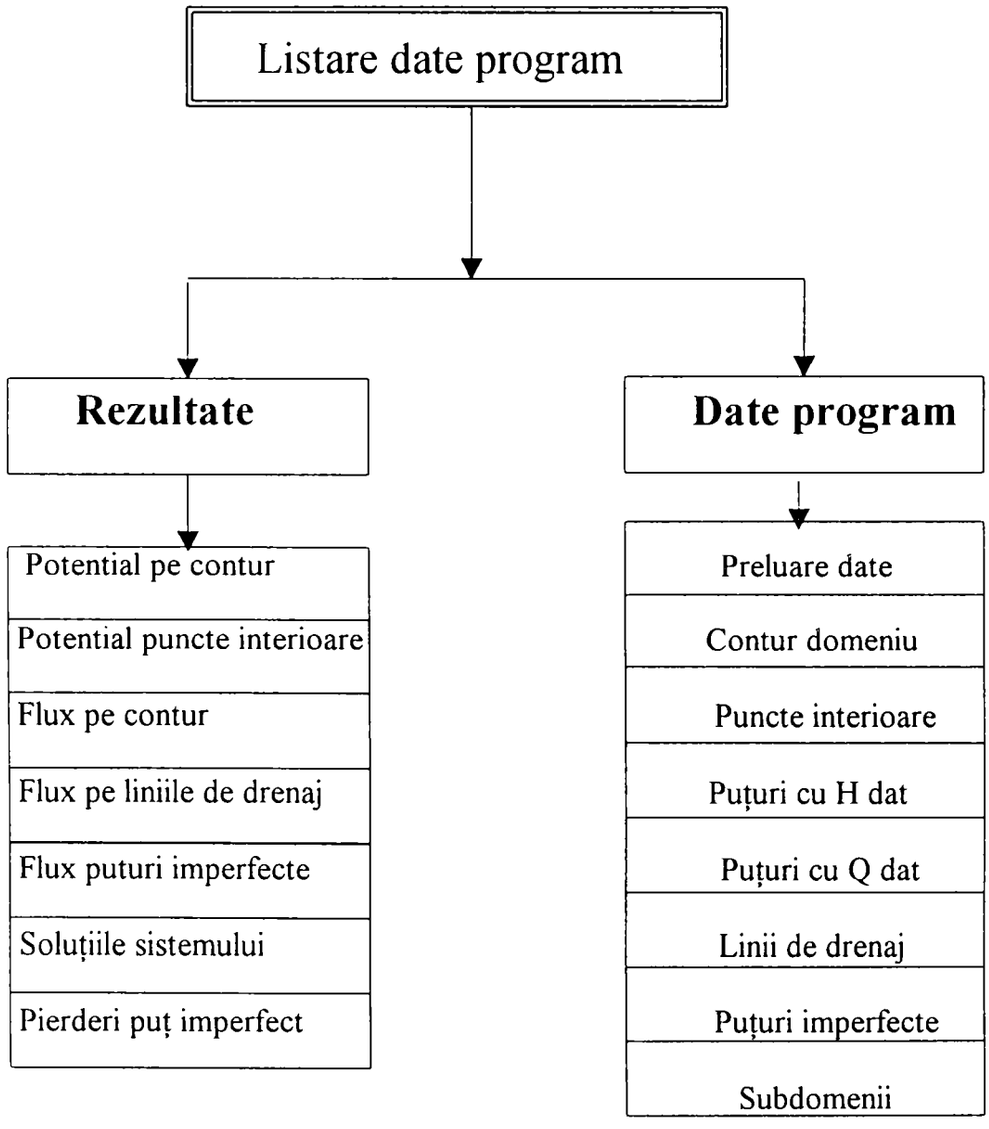


Fig. 7.14 Prezentarea meniului Listare date program din cadrul programului MEFRO

APLICAȚII ALE PROGRAMULUI MEFRO ÎN CADRUL SISTEMELOR COMPLEXE ALE CAPTĂRILOR SUBTERANE

În vederea testării programului elaborat s-au conceput şi rulat o serie de exemple standard pentru care în mare parte a existat posibilitatea comparării cu rezultate exacte.

Ex.1 **Calculul debitului unui puț izolat având o denivelare constantă.**

În cadrul acestui exemplu s-a luat în considerare un puț situat într-un domeniu circular cu raza de influență de 10 m. raza puțului fiind 0,1m. (fig. 8.1). S-a considerat realizată o denivelare de 1m. Discretizarea domeniului este prezentată în tabelul 8.1

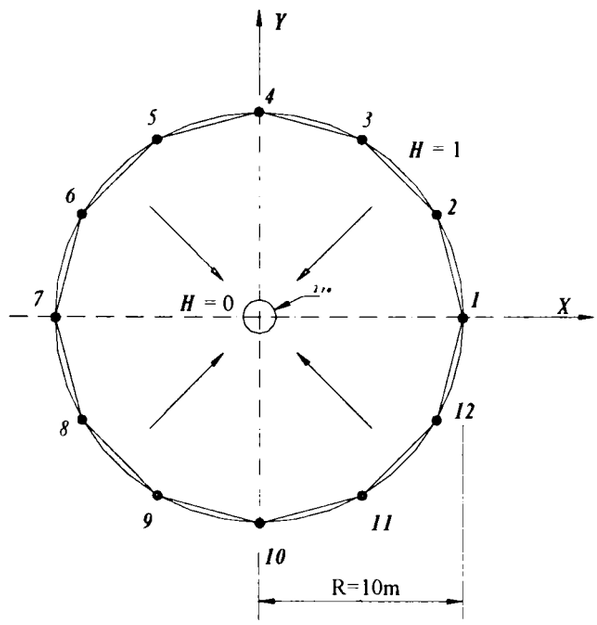


Fig.8.1 Discretizarea frontierei în cazul unui puț cu denivelare constantă

Valoarea debitului calculat analitic se obține cu ajutorul formulei cunoscute a puțurilor:

$$Q_A = \frac{2\pi kmS}{\ln \frac{R}{r_0}}$$

unde semnificația mărimilor care intervin este următoarea:

- Q_A debitul puțului calculat analitic;
- km transmisivitatea domeniului;
- R raza domeniului circular;
- r_0 raza puțului;
- S denivelarea realizată.

De asemenea au fost calculate valorile fluxului şi a potențialului în anumite puncte din interiorul domeniului, atât numeric cât şi analitic pentru a putea fi comparate în vederea stabilirii erorii cu care metoda de calcul numerică (MEFRO) furnizează valorile acestor mărimi. Aceste rezultate sunt prezentate în tabelele 8.2.

Tabelul 8.1

Nr. nod	Abscisă (m)	Ordonată (m)	Potențial contur (m)
1	10	0	1,00
2	8.7	5	1,00
3	5	8.7	1,00
4	0	10	1,00
5	-5	8.7	1,00
6	-8.7	5	1,00
7	-10	0	1,00
8	-8.7	-5	1,00
9	-5	-8.7	1,00
10	0	-10	1,00
11	5	-8.7	1,00
12	8.7	-5	1,00

Tabelul 8.2

Nr. element	H(m) MEFRO	Flux (q/l) MEFRO	H(m) analitic	Flux(q/l) analitic
1	1,00	-0,0217	1,00	-0.0217
2	1,00	-0,0218	1,00	-0.0217
3	1,00	-0,0217	1,00	-0.0217
4	1,00	-0,0217	1,00	-0.0217
5	1,00	-0,0218	1,00	-0.0217
6	1,00	-0,0217	1,00	-0.0217
7	1,00	-0,0217	1,00	-0.0217
8	1,00	-0,0218	1,00	-0.0217
9	1,00	-0,0217	1,00	-0.0217
10	1,00	-0,0217	1,00	-0.0217
11	1,00	-0,0218	1,00	-0.0217
12	1,00	-0,0218	1,00	-0.0217

Observație:

Valoarea fluxului total (debitul puțului) calculat analitic este: Q = - 1,364

Valoarea fluxului (debitului) calculat cu MEFRO : Q = -1,373

Eroarea de calcul a debitului (global)este : $\varepsilon_{Q\%} = \left| \frac{Q_{MEFRO} - Q_{Analitic}}{Q_{Analitic}} \right| = 0{,}66\%$

Ex.2 **Calculul denivelării într-un puț izolat cunoscând debitul constant extras**

În acest caz s-a urmărit compararea denivelării calculate folosind metoda elementelor de frontieră respectiv programul MEFRO cu soluția analitică. În cadrul acestui caz s-a luat în considerare un puț cu raza de influență de 100 m, raza puțului fiind considerată 0,1m (fig. 7.1). S-a considerat extragerea unui debit cu valoarea de 0,909. Calculând analitic denivelarea corespunzătoare acestui debit rezultă o valoare de 1,000 m.

Schița discretizării este prezentată în figura 8.2 iar discretizarea domeniului împreună cu condițiile de margine sunt prezentate în tabelul 8.3.

În tabelul 8.4. sunt prezentate valorile potențialului şi a fluxului pe conturul domeniului calculate cu MEFRO, comparativ cu valorile calculate analitic.

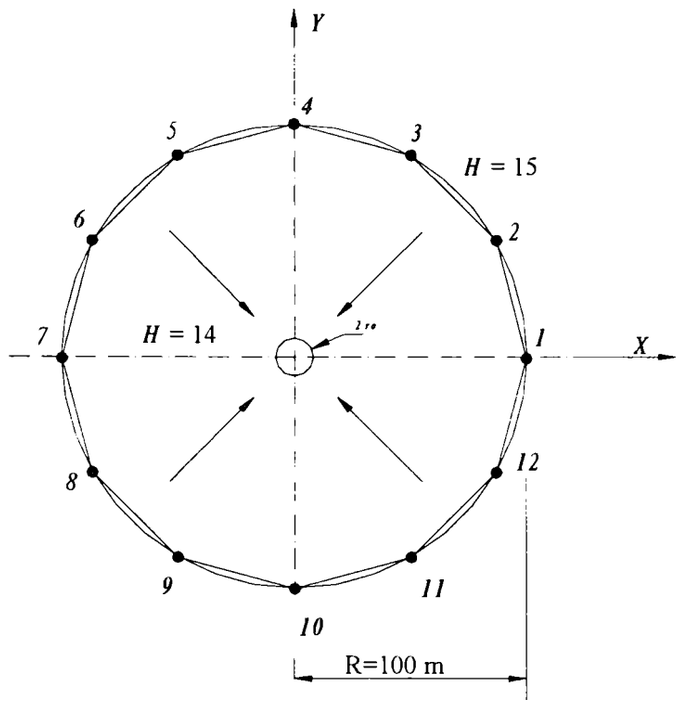


Fig 8.2 Discretizarea frontierei în cazul unui puț cu debit constant.

Tabelul 8.3

Nr. nod	Abscisă (m)	Ordonată (m)	Potențial contur (m)
1	100,0	0,00	15,00
2	87,00	50,00	15,00
3	50,00	87,00	15,00
4	0,000	100,0	15,00
5	-50,0	87,00	15,00
6	-87,0	50,00	15,00
7	-100	0,000	15,00
8	-87,0	-50,0	15,00
9	-50,0	-87,0	15,00
10	0,00	-10,0	15,00
11	50,0	-87,0	15,00
12	87,0	-50,0	15,00

Tabelul 8.4

Nr. element	H(m) MEFRO	Flux (q/l) MEFRO	H(m) analitic	Flux(q/l) Analitic
1	15,00	-0.00142	15,00	-0.00144
2	15,00	-0.00143	15,00	-0.00144
3	15,00	-0.00142	15,00	-0.00144
4	15,00	-0.00142	15,00	-0.00144
5	15,00	-0.00143	15,00	-0.00144
6	15,00	-0.00142	15,00	-0.00144
7	15,00	-0.00142	15,00	-0.00144
8	15,00	-0.00143	15,00	-0.00144
9	15,00	-0.00142	15,00	-0.00144
10	15,00	-0.00142	15,00	-0.00144
11	15,00	-0.00143	15,00	-0.00144
12	15,00	-0.00142	15,00	-0.00144

Valoarea înălțimii piezometrice calculată în puț este:

$H_{put}^{MEFRO} = 13{,}954\ \text{m}$

Valoarea înălțimii piezometrice în puț calculată analitic este:

$H_{put}^{MEFRO} = 14{,}000\ \text{m}$

Eroarea de calcul a înălțimii piezometrice în puț este:

$$\varepsilon_{H\%} = \left| \frac{H_{MEFRO} - H_{Analitic}}{H_{Analitic}} \right| = 0{,}32\%$$

În acest caz, pentru stabilirea erorii de calcul a programului MEFRO față de soluția analitică s-a luat în considerare o linie de drenaj cu lungimea de 50 m, situată într-un domeniu având frontiera o elipsă cu raza mare R=100. Potențialul pe conturul exterior a fost considerat $H_{ext} = 15\,\text{m}$, iar cel de pe linia de drenaj $H_{dren} = 10\,\text{m}$.

Schița discretizării frontierei şi a liniei de dren este prezentată în figura 8.3

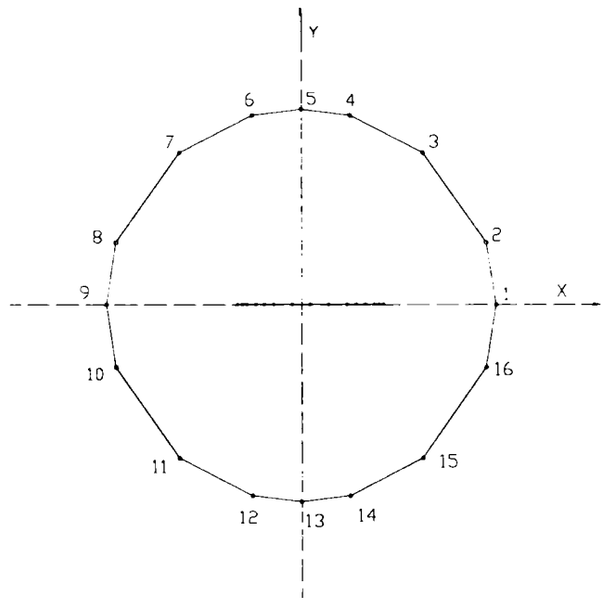


Fig. 8.3 Discretizarea frontierei şi a liniei de dren în cazul unui dren de lungime finită

Tabel 8.5

Nod.crt linie dren	Abscisă [m]	Ordonată [m]
1	-25.00	0.00
2	-23.45	0.00
3	-21.88	0.00
4	-18.75	0.00
5	-15.70	0.00
6	-12.50	0.00
7	-6.25	0.00
8	0.00	0.00
9	6.25	0.00
10	12.50	0.00
11	15.70	0.00
12	18.75	0.00
13	21.88	0.00
14	23.45	0.00
15	25.00	0.00

Tabel 8.6

Nr. element dren	Potential elem. dren	$\frac{q}{km}$	$\frac{q}{km} l_e$
1	10.00	1.0435	1.6174
2	10.00	0.4596	0.7216
3	10.00	0.3395	1.0627
4	10.00	0.2666	0.8132
5	10.00	0.2353	0.7531
6	10.00	0.2083	1.3021
7	10.00	0.1934	1.2089
8	10.00	0.1934	1.2089
9	10.00	0.2083	1.3021
10	10.00	0.2353	0.7531
11	10.00	0.2666	0.8132
12	10.00	0.3395	1.0627
13	10.00	0.4596	0.7216
14	10.00	1.0435	1.6174

l_e - lungimea elementului de dren

Valoarea fluxului total al liniei de dren calculată cu MEFRO este:

$$\frac{Q_{dren}^{MEFRO}}{km} = 14{,}958$$

Valoarea fluxului liniei de dren calculată analitic [21] este:

$$\frac{Q_{dren}^{Analitic}}{km} = \frac{2\pi(H_C - H_{dren})}{\ln\left(\frac{R}{l_d/2} + \sqrt{\frac{R^2}{l_d/2} + 1}\right)} = 14{,}997 \text{ m}$$

Eroarea de calcul a fluxului total (debitului) liniei de dren este:

$$\varepsilon_{Q\%} = \left|\frac{Q_{MEFRO} - Q_{Analitic}}{Q_{Analitic}}\right| = 0{,}26\%$$

Pentru evaluarea erorii metodei elementelor de frontieră cu programul MEFRO în cazul puțurilor imperfecte s-au luat în considerare relațiile analitice obținute de Muskat şi Huisman.

Relația de calcul obținută de Muskat are forma:

$$\frac{Q}{k_f m} = \frac{2\pi(H_R - H_0)}{ln\frac{R}{r_0} + \left(\frac{m}{l} - 1\right) ln\frac{4m}{r_0} - \frac{m}{2l} f\left(\frac{l}{m}\right)} \tag{8.1}$$

unde valorile funcției f sunt exprimate în tabelul 8.7

Tabelul 8.7

l/m	0.05	0.08	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f(l/m)	8	7	6.5	5.2	4.2	3.6	3	2.4	1.9	1.4	0.8	0

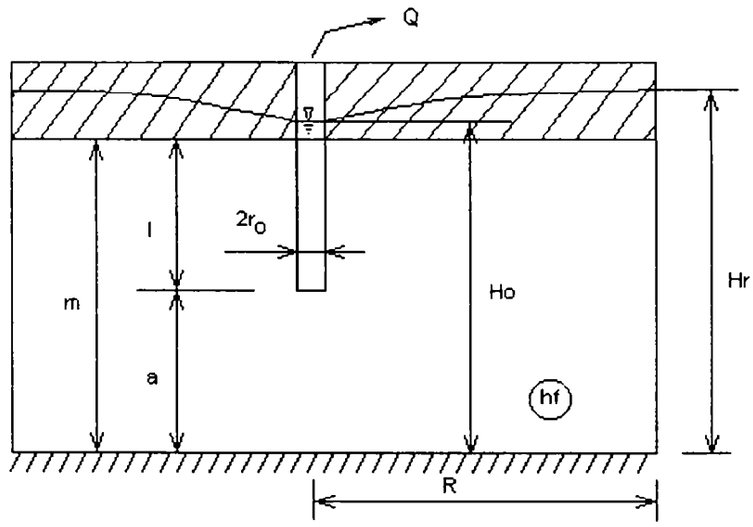


Fig. 8.4 Schița unui puț imperfect

O altă formulă pentru calculul debitului puțului imperfect este cea a lui Huisman :

$$\frac{Q}{k_f m} = \frac{2\pi(H_R - H_0)}{ln\frac{R}{r_0} + \left(\frac{m}{l} - 1\right) ln\left(\alpha\frac{l}{r_0}\right)} \tag{8.2}$$

unde valorile coeficientului α sunt prezentate în tabelul 8.8 funcție de următorii parametrii:

$$e = \frac{1}{2}\left(1 - \frac{l}{m}\right) \quad \text{şi} \quad p = \frac{l}{m}$$

Valorile coeficientului α Tabelul 8.8

	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45
0.1	**0.54**	**0.55**	**0.55**	**0.56**	**0.57**	**0.59**	**0.61**	**0.67**	**1.09**
0.2	**0.44**	**0.45**	**0.46**	**0.47**	**0.49**	**0.52**	**0.59**	**0.89**	
0.3	**0.37**	**0.38**	**0.39**	**0.41**	**0.43**	**0.50**	**0.74**		
0.4	**0.31**	**0.32**	**0.34**	**0.36**	**0.42**	**0.62**			
0.5	**0.36**	**0.27**	**0.29**	**0.34**	**0.51**				
0.6	**0.21**	**0.23**	**0.27**	**0.41**					
0.7	**0.17**	**0.20**	**0.32**						
0.8	**0.13**	**0.22**							
0.9	**0.12**								

Analizând formulele de calcul a debitului prezentate mai sus se observă că ele au o structură similară. Termenul suplimentar înmulțit cu $(m/l-1)$ sau m/l reprezintă efectul imperfecțiunii puțului care poate fi interpretat ca o pierdere de sarcină suplimentară datorită imperfecțiunii. Dezavantajul major al acestor formule este faptul că ele folosesc coeficienți din tabele şi au o valabilitate limitată pentru puțul singular în mişcare axial simetrică. De asemenea aceste formule nu permit determinarea distribuției de debit de-a lungul puțului.

Tabelele 8.9 şi 8.10 prezintă rezultatele comparative obținute cu formulele de mai sus menționate.

Valori ale debitului calculate cu formula lui **Muskat** Tabelul 8.9

	R=100m, r_0=0.125m			R=100m, r_0=0.25m			R=100m, r_0=0.5m		
R/r_0	800			400			200		
l/m	**12/20**	**8/20**	**6/20**	**12/20**	**8/20**	**6/20**	**12/20**	**8/20**	**6/20**
f(l/m)	**2.4**	**3.6**	**4.2**	**2.4**	**3.6**	**4.2**	**2.4**	**3.6**	**4.2**
Q/(KT)	1.397	1.058	0.851	1.603	1.238	1.009	1.880	1.494	1.239

Valori ale debitului calculate cu formula lui **Huisman** Tabelul 8.10

R/r_0	R=100m, r_0=0.125m 800			R=100m, r_0=0.25m 400			R=100m, r_0=0.5m 200		
E	**0.2**	**0.3**	**0.35**	**0.2**	**0.3**	**0.35**	**0.2**	**0.3**	**0.35**
P	**0.6**	**0.4**	**0.3**	**0.6**	**0.4**	**0.3**	**0.6**	**0.4**	**0.3**
α	**0.41**	**0.62**	**0.74**	**0.41**	**0.62**	**0.74**	**0.41**	**0.62**	**0.74**
Q/KT	1.375	1.029	0.836	1.575	1.199	0.989	1.841	1.437	1.209

Valori ale debitului calculate cu MEFRO(M.E.A.) Tabelul 8.11

	R=100m, r_o=0.125m R/r0 = 800			R=100m, r_o=0.25m R/r0 = 400			R=100m, r_o=0.5m R/r_0 = 200		
$l_{element}$ (m)	**2.00**	**1.33**	**1.00**	**2.00**	**1.33**	**1.00**	**2.00**	**1.33**	**1.00**
q_1/KT	**0.278**	**0.217**	**0.182**	**0.334**	**0.275**	**0.240**	**0.426**	**0.383**	**0.356**
q_2/KT	**0.222**	**0.173**	**0.140**	**0.262**	**0.193**	**0.161**	**0.290**	**0.221**	**0.171**
q_3/KT	**0.222**	**0.165**	**0.133**	**0.252**	**0.190**	**0.155**	**0.288**	**0.222**	**0.185**
q_4/KT	**0.216**	**0.160**	**0.130**	**0.246**	**0.183**	**0.145**	**0.280**	**0.213**	**0.174**
q_5/KT	**0.214**	**0.158**	**0.127**	**0.242**	**0.180**	**0.146**	**0.276**	**0.207**	**0.171**
q_6/KT	**0.214**	**0.157**	**0.126**	**0.240**	**0.178**	**0.145**	**0.274**	**0.206**	**0.170**
$\Sigma Q_i/KT$	1.366	1.030	0.838	1.576	1.199	0.992	1.834	1.452	1.227

Se poate observa faptul că valorile calculate cu MEFRO (M.E.A.) sunt foarte apropiate de valorile calculate cu relația lui Huisman [8.2]. De asemenea în cele mai multe din cazuri când $4 < l_{elem}/r_0 < 16$ valorile calculate ale debitului sunt foarte apropiate cu cele obținute aplicând relația de calcul a lui Huisman (8.2). sau chiar identice. Totuşi diferențele obținute aplicând MEFRO (MEA) pentru calculul debitului puțurilor imperfecte raportate la relația de calcul a lui Huisman nu le depăşesc pe cele obținute prin aplicarea relației lui Muskat (8.1).

O imagine de ansamblu asupra rezultatelor obținute este prezentată în tabelul 8.12.

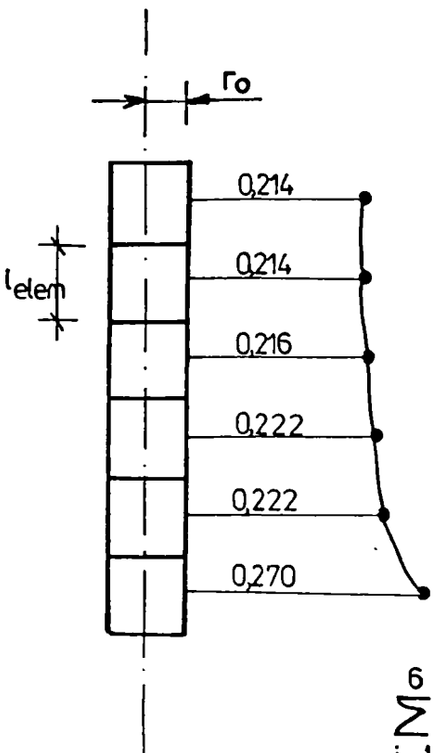
Tabelul 8.12

	Valorile calculate ale debitului								
	R=100m , ro=0,125m , R/ro=800			R=100m , ro=0,25m , R/ro=800			R=100m , ro=0,5m , R/ro=800		
L/m	12/20	8/20	6/20	12/20	8/20	6/20	12/20	8/20	6/20
Muskat	1.397	1.058	0.851	1.603	1.238	1.009	1.880	1.494	1.239
Huisman	1.375	1.029	0.836	1.575	1.199	0.989	1.841	1.437	1.209
MEFRO	1.366	1.030	0.838	1.576	1.199	0.992	1.834	1.452	1.227

Luând în considerare exemplul de calcul referitor la erorile introduse de puțurile imperfecte folosind programul de simulare numerică MODFLOW, prezentat în cadrul capitolului IV, se observă acuratețea incontestabilă oferită de MEFRO (MEA). Pentru o mai bună relevanță s-a rulat folosind programul MEFRO un exemplu de calcul care are aceleaşi condiții cu cele de la exemplul folosit în cazul aplicării MODFLOW. Rezultatele comparative sunt prezentate în tabelul 8.13.

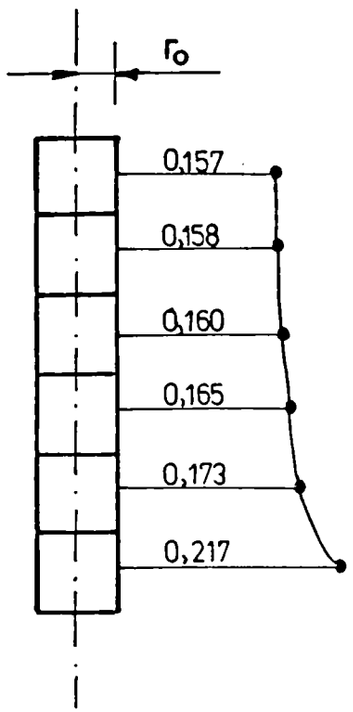
$R_{ext} = 100\,m$, $r_o = 0{,}125\,m$, $m = 20\,m$, $H_R = 24\,m$, $H_{put} = 22\,m$

$l_{elem} = 2{,}00\,m$
$l_{put} = 12\,m$



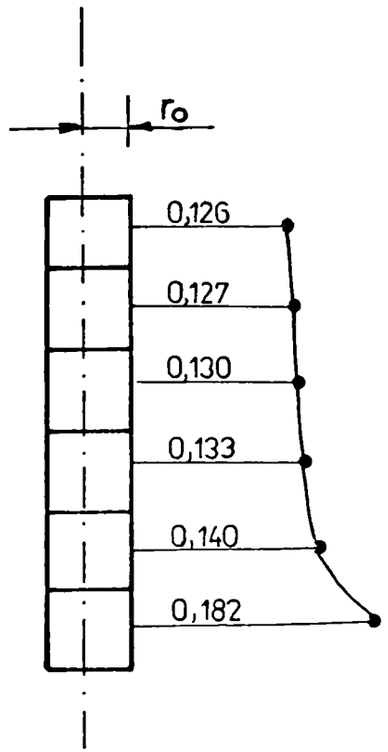
$$\sum_{i=1}^{6} \frac{q_i}{KT} = 1{,}366$$

$l_{elem} = 1{,}33\,m$
$l_{put} = 8\,m$



$$\sum_{i=1}^{6} \frac{q_i}{KT} = 1{,}030$$

$l_{elem} = 1{,}00\,m$
$l_{put} = 6\,m$

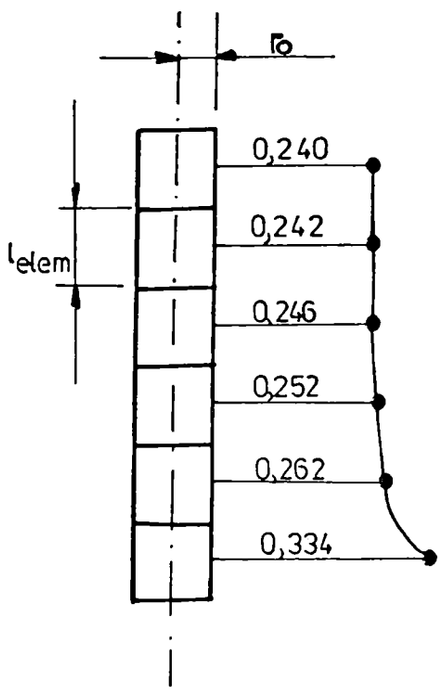


$$\sum_{i=1}^{6} \frac{q_i}{KT} = 0{,}838$$

Fig. 8.5

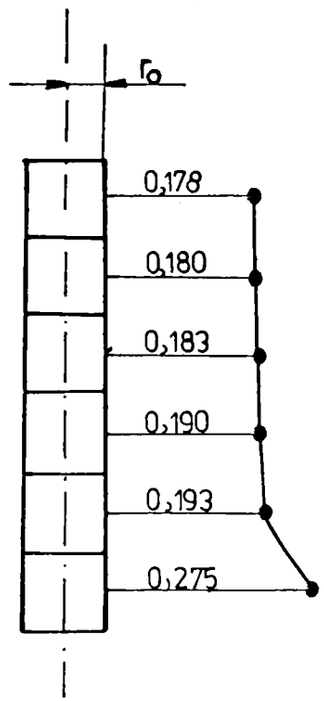
R_{ext} = 100 m , r_o = 0 25 m , m = 20m , H_R = 24 m , H_{put} = 22 m

l_{elem} = 2,00 m
l_{put} = 12,00 m



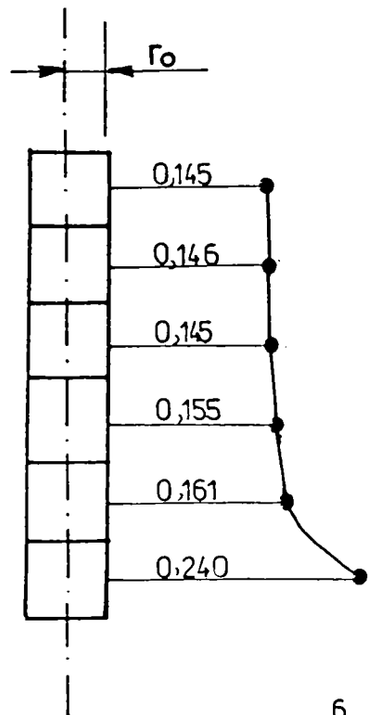
$$\sum_{i=1}^{6} \frac{q_i}{KT} = 1,576$$

l_{elem} = 1 33 m
l_{put} = 8,00 m



$$\sum_{i=1}^{6} \frac{q_i}{KT} = 1,199$$

l_{elem} = 1,00 m
l_{put} = 6,00 m

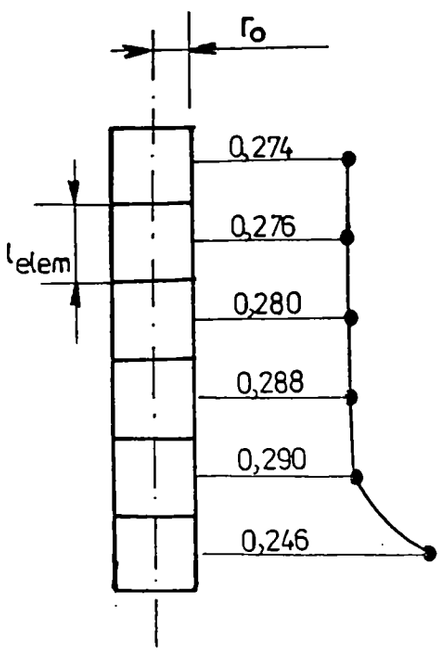


$$\sum_{i=1}^{6} \frac{q_i}{KT} = 0,992$$

Fig. 8.6

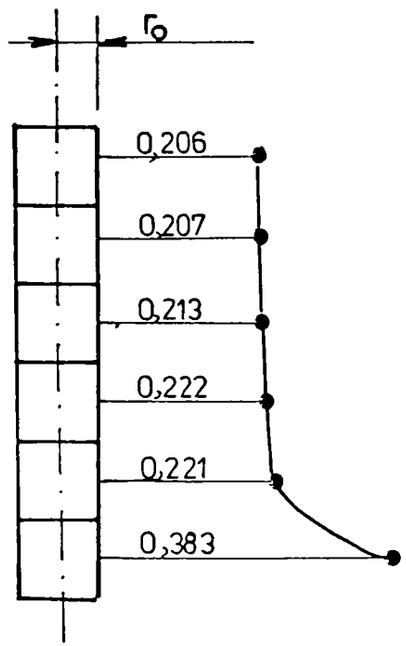
$R_{ext} = 100\,m$, $r_o = 0{,}5\,m$, $m = 20\,m$, $H_R = 24\,m$, $H_{puț} = 22\,m$

$l_{elem} = 2{,}00\,m$
$l_{puț} = 12{,}00\,m$



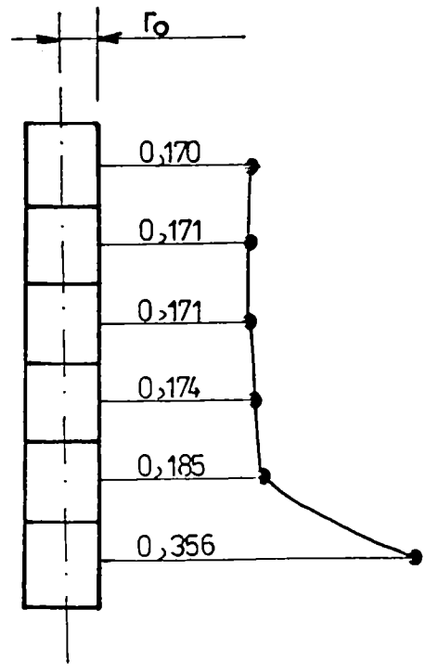
$$\sum_{i=1}^{6} \frac{q_i}{KT} = 1834$$

$l_{elem} = 1{,}33\,m$
$l_{puț} = 8{,}00\,m$



$$\sum_{i=1}^{6} \frac{q_i}{KT} = 1452$$

$l_{elem} = 1{,}00\,m$
$l_{puț} = 6{,}00\,m$



$$\sum_{i=1}^{6} \frac{q_i}{KT} = 1227$$

Fig. 8.7

Tabelul 8.13

MODFLOW					MEA / MEFRO					MUSKAT
H_{put} [m]	Nr. straturi	Discretizare plana	Q [m^3/zi]	ε_Q [%]	l_{elem} [m]	Nr. elem	ql_{elem}/ km	Q [m^3/zi]	ε_Q [%]	Q [m^3/zi]
4	5	10x10	1176.3	145	4	1	0.900	432.34	9.70	479.1
		5x5	812.9	69.6						
		2x2	528.6	10.3						
		1x1	406.9	15.0						
	10	10x10	1192.2	148	2	1	0.548	447.71	6.55	
		5x5	836.4	74.5		2	0.384			
		2x2	552.7	15.36						
		1x1	431.5	9.93						
	15	10x10	1195.7	149	1.3	1	0.432	455.402	4.94	
		5x5	841.5	75.6		2	0.254			
		2x2	560.6	17.0		3	0.260			
		1x1	429.9	10.2						
8	5	10x10	1499.3	109	2	1	0.786	679.740	5.28	717.7
		5x5	1117.2	55.6		2	0.628			
		2x2	799.1	11.3						
		1x1	647.9	9.72						
	10	10x10	1510.7	110	4	1	0.484	693.67	3.34	
		5x5	1134.5	58.0		2	0.328			
		2x2	817.8	13.9		3	0.320			
		1x1	665.8	7.23		4	0.312			
	15				6	1	0.384	699.919	2.47	
		10x10	1513.2	110		2	0.221			
		5x5	1138.2	58.5		3	0.223			
		2x2	823.9	14.8		4	0.213			
		1x1	668.7	6.82		5	0.208			
						6	0.208			
12	5	10x10	1723.4	90.7	4	1	0.713	880.54	2.54	903.5
		5x5	1342.3	48.5		2	0.568			
		2x2	1011.8	11.9		3	0.552			
		1x1	845.9	6.37						
	10				2	1	0.438	891.60	1.31	
		10x10	1731.6	91.65		2	0.298			
		5x5	1355.2	49.98		3	0.290			
		2x2	1026.6	13.62		4	0.280			
		1x1	857.3	5.11		5	0.276			
						6	0.274			
	15				1	1	0.306	900.24	0.30	
		10x10	1733.5	91.86		2	0.147			
						3	0.159			
		5x5	1358.1	50.31		4	0.148			
						5	0.145			
		2x2	1031.6	14.17		6	0.142			
						7	0.140			
		1x1	862.4	4.54		8	0.139			
						9	0.138			
						10	0.137			
						11	0.137			
						12	0.136			

Analizând datele din tabelul 8.13 se desprind următoarele concluzii:

Referitor la aplicarea programului de calcul MODFLOW pentru puțuri imperfecte

- erorile relative de calcul scad pe măsură ce gradul de penetrare a puțului creşte, ceea ce ne conduce la concluzia că cu cât ne apropiem de situația unui puț perfect erorile relative se reduc, ele rămânând însă semnificative;
- pentru a se putea obține erori de calcul de sub 10% se constată că este nevoie de o discretizare de cel puțin $1x1m$ chiar şi în această situație pentru un raport $m/H_{put} \geq 5$ se constată erori care depăşesc această valoare;
- pentru obținerea unei erori de calcul de sub 5% este necesar ca raportul $m/H_{put} \leq 2$ împreună cu o discretizare de cel puțin $1x1x15m$;
- folosirea diferitelor grade de discretizare în plan vertical nu influențează semnificativ eroarea relativă de calcul;

Referitor la aplicarea programului MEFRO:

- comparativ cu orice situație folosită în cadrul programului MODFLOW erorile relative de calcul a debitului sunt mai mici de 10%;
- şi în cazul reprezentării puțului imperfect printr-un singur element eroarea relativă de calcul a debitului este de sub 10%;
- şi în acest caz erorile de calcul scad pe măsură ce gradul de penetrare a puțului creşte;
- la o reducere a raportului m/H_{put} de la valoarea $m/H_{put} = 5$ la valoarea $m/H_{put} = 1.66$ păstrând acelaşi grad de discretizare al puțului imperfect eroarea relativă de calcul a debitului se reduce cu aproape 50%;
- față de MODFLOW programul MEFRO calculează şi distribuția de debit pe elementele puțului imperfect oferind astfel o imagine mult mai clară a fluxului distribuit de-a lungul puțului imperfect.

Concluziile menționate mai sus conduc la prin aplicare MEFRO/MEA pentru calculul puțurilor imperfecte se obțin rezultate net superioare comparativ cu rezultatele obținute de către programe care au deja o largă folosire pe plan mondial (MODFLOW). Volumul necesar al datelor de intrare este substanțial redus în cazul MEFRO/MEA ceea ce conduce şi la un timp redus de rezolvare a acestui gen de probleme prin reducerea timpului necesar pregătirii datelor de intrare.

Ex.5 **Cazul modelării aportului din precipitații**

Pentru testarea programului MEFRO în cazul modelării aportului din precipitații s-a luat în studiu un domeniu simplu de formă pătrată. S-a considerat cazul când aportul din precipitații este distribuit pe toată suprafața domeniului şi se infiltrează în totalitate în stratul acvifer sub presiune.

Pentru testarea modelului s-a apelat la calculul înălțimii piezometrice în interiorul domeniului folosind soluția exactă cunoscută [22]:

$$h_{(x)} = H_0 + \frac{\varepsilon L^2}{8km}\left(1 - 4\frac{x^2}{L^2}\right)$$

Schița discretizării frontierei domeniului şi a subdomeniului, împreună cu condițiile de margine este prezentată în fig 8.8.

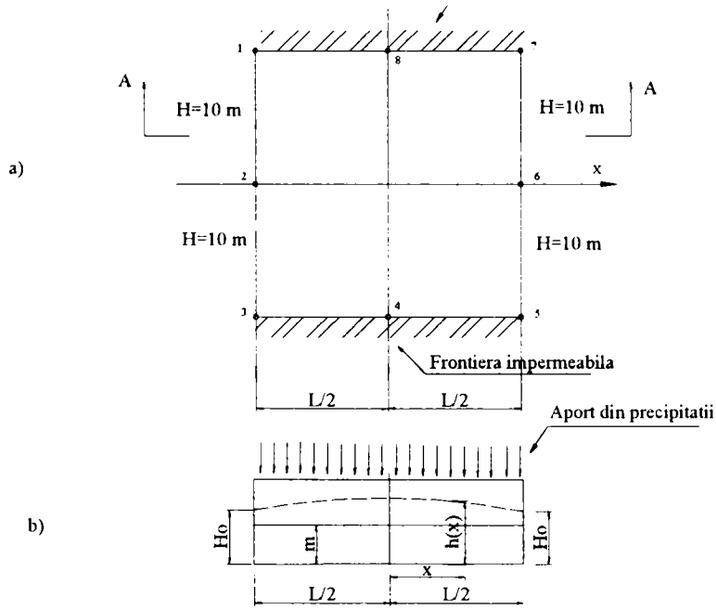


Fig.8.8 Schița exemplului pentru:
a) vedere în plan, discretizare, condiții la limită;
b) secțiune longitudinală

Exemplul numeric considerat este un pătrat cu latura de 100 m. S-a considerat un aport de precipitații de 300 mm/an. Transmisivitatea s-a considerat ca fiind $km = 10^{-3}$. Elementele 1, 2; 5, 6 de frontiera sunt considerate ca fiind elemente de frontieră cu potențial cunoscut $H_0 = 10m$, iar elementele 3, 4, 7, 8 sunt considerate cu flux nul (impermeabile), conform figurii 8.8a. Datele pentru discretizarea domeniului sunt prezentate în tabelul 8.14. Rezultatele comparative obținute în urma rulării MEFRO sunt prezentate în tabelul 8.15.

Tabelul 8.14

Nr punct	Abscisă [m]	Ordonată [m]	Potențial [m]
1	-50,00	50,00	10,00
2	-50,00	0,000	10,00
3	-50,00	-50,00	0,000
4	0,000	-50,00	0,000
5	50,00	-50,00	10,00
6	50,00	0,000	10,00
7	50,00	50,00	0,000
8	0,000	50,00	0,000

Tabelul 8.15

Nr.punct interior	Abscisă [m]	Ordonată [m]	Potențial calculat analitic [m]	Potențial calculat MEFRO [m]
1	-50,00	0,000	10,0000	9,9994
2	-25,00	0,000	10,0093	10,0094
3	0,000	0,000	10,0125	10,0126
4	25,00	0,000	10,0093	10,0094
5	50,00	0,000	10,0000	9,9994

După cum se poate observa din tabelul 8.15 precizia de calcul a programului MEFRO este de ordinul milimetrilor deci sub 1%, ceea ce îi confirmă corectitudinea şi acuratețea.

În general se pot lua în considerare în cadrul programului subdomenii mai mici incluse în domeniul mişcării, aşa după cum s-a putut observa la prezentarea generală a programului.

Ex.6 **Cazul puțurilor imperfecte înnisipate, cu luarea în considerare a pierderilor de sarcină interioare.**

Pentru verificarea aplicabilității programului MEFRO în cazul puțurilor imperfecte când se consideră efectul mişcării interioare, cu pierderi de sarcină, s-a considerat un puț de infiltrare imperfect, cu raza de 0,5m, un coeficient de filtrație pentru nisipul din interiorul puțului $k_0 = 0.01m/s$ iar pentru domeniul exterior (acvifer) $k_f = 3.927 \cdot 10^{-5}$. Raza de influență s-a considerat $R = 100m$. Puțul de infiltrație folosit de exemplu pentru îmbogățirea artificială a acviferului are drept condiții de margine $H_P = 22m.$ în puț, iar potențialul pe conturul domeniului $H_{ext} = 20m$. Discretizarea frontierei domeniului are 12 elemente (fig.8.9).

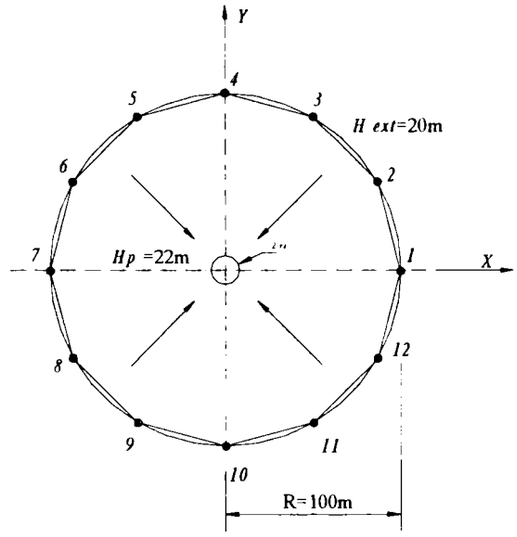


Fig. 8.9 Discretizarea frontierei domeniului pt. puț imperfect înisipat

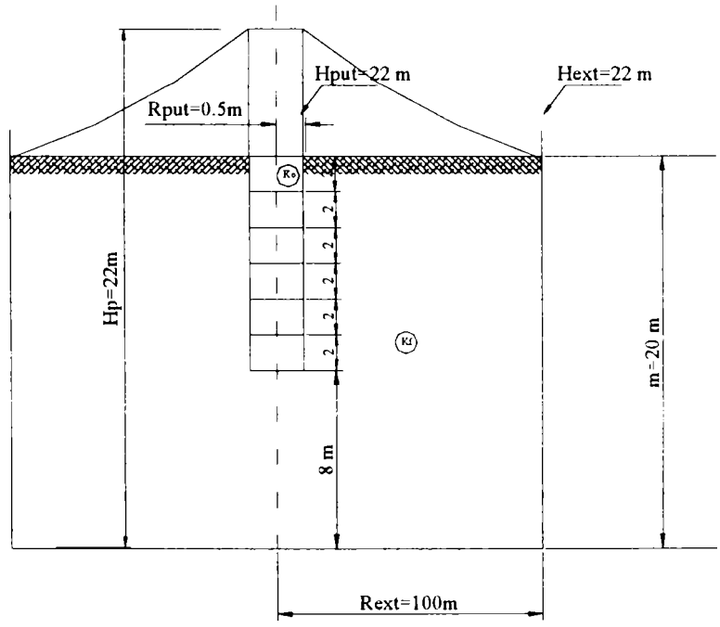


Fig. 8.10 Discretizarea puțului imperfect înnisipat în plan vertical

Discretizarea puțului imperfect s-a realizat în plan vertical cu 6 elemente de lungime $l_e = 2m$ fiecare. Grosimea stratului acvifer a fost considerată $m = 20m$, parametrul imperfecțiunii având valoarea de 8m (fig.8.10).

Aşa după cum s-a precizat la prezentarea generală în cadrul capitolului V luarea în considerare a pierderilor de sarcină în coloana de nisip din interior implică iterații.

Rularea programului a necesitat un număr de 5 iterații până la îndeplinirea condiției:

$$\frac{\left|H_i^{elem} - H_{i+1}^{elem}\right|}{H_1^{elem}} < 0.01m$$

adică până când diferența dintre valorile înălțimii piezometrice pe un element între două iterații succesive este de sub 1%.

Semnificația denumirilor utilizate în fişierul de rezultate este următoarea:

H_puț_precedent – valoarea potențialului în puț pe element la iterația precedentă;
H_puț_calculat – valoarea înălțimii piezometrice calculate pe element după iterația actuală;
Delta h *i* - reprezintă pierderea de sarcină de-a lungul elementului „*i*" corespunzătoare iterației;
Fluxul *i* - este fluxul exfiltrat pe element corespunzător iterației.

În figura 8.11 se poate observa influența pe care o poate avea luarea în considerare a pierderilor de sarcină din interiorul puțului.

Rezultatele rulării sunt prezentate în cele ce urmează:

```
Iteratia nr.  1

Fluxul total pe putul imperfect 1 =    -1.8586
H_put_precedent =    22.0000
H_put_calc      =    21.8279
Delta h 6       =     0.3443
Fluxul  6       =    -0.1373

H_put_precedent =    22.0000
H_put_calc      =    21.5112
Delta h 5       =     0.2891
Fluxul  5       =    -0.1382

H_put_precedent =    22.0000
H_put_calc      =    21.2499
Delta h 4       =     0.2334
Fluxul  4       =    -0.1403

H_put_precedent =    22.0000
H_put_calc      =    21.0449
Delta h 3       =     0.1764
Fluxul  3       =    -0.1447
```

```
H_put_precedent =    22.0000
H_put_calc      =    20.8979
Delta h 2       =     0.1177
Fluxul  2       =    -0.1493

H_put_precedent =    22.0000
H_put_calc      =    20.8171
Delta h 1       =     0.0439
Fluxul  1       =    -0.2195



Iteratia nr.  2

Fluxul total pe putul imperfect 1 =    -1.0998
H_put_precedent =    21.8279
H_put_calc      =    21.9086
Delta h 6       =     0.1827
Fluxul  6       =    -0.1862

H_put_precedent =    21.5112
H_put_calc      =    21.7562
Delta h 5       =     0.1222
Fluxul  5       =    -0.1165

H_put_precedent =    21.2499
H_put_calc      =    21.6540
Delta h 4       =     0.0821
Fluxul  4       =    -0.0838

H_put_precedent =    21.0449
H_put_calc      =    21.5863
Delta h 3       =     0.0533
Fluxul  3       =    -0.0604

H_put_precedent =    20.8979
H_put_calc      =    21.5436
Delta h 2       =     0.0322
Fluxul  2       =    -0.0450

H_put_precedent =    20.8171
H_put_calc      =    21.5217
Delta h 1       =     0.0116
Fluxul  1       =    -0.0579
```

```
Iteratia nr.  3

Fluxul total pe putul imperfect 1 =    -1.5309
H_put_precedent =    21.9086
H_put_calc      =    21.8627
Delta h 6       =     0.2746
Fluxul  6       =    -0.1578

H_put_precedent =    21.7562
H_put_calc      =    21.6161
Delta h 5       =     0.2185
Fluxul  5       =    -0.1228

H_put_precedent =    21.6540
H_put_calc      =    21.4212
Delta h 4       =     0.1715
Fluxul  4       =    -0.1124

H_put_precedent =    21.5863
H_put_calc      =    21.2718
Delta h 3       =     0.1273
Fluxul  3       =    -0.1083

H_put_precedent =    21.5436
H_put_calc      =    21.1660
Delta h 2       =     0.0842
Fluxul  2       =    -0.1075

H_put_precedent =    21.5217
H_put_calc      =    21.1082
Delta h 1       =     0.0313
Fluxul  1       =    -0.1567


Iteratia nr.  4

Fluxul total pe putul imperfect 1 =    -1.2799
H_put_precedent =    21.8627
H_put_calc      =    21.8894
Delta h 6       =     0.2211
Fluxul  6       =    -0.1744

H_put_precedent =    21.6161
H_put_calc      =    21.6977
Delta h 5       =     0.1623
Fluxul  5       =    -0.1195

H_put_precedent =    21.4212
H_put_calc      =    21.5570
Delta h 4       =     0.1192
Fluxul  4       =    -0.0962
```

```
H_put_precedent =    21.2718
H_put_calc      =    21.4555
Delta h 3       =     0.0838
Fluxul  3       =    -0.0805

H_put_precedent =    21.1660
H_put_calc      =    21.3868
Delta h 2       =     0.0536
Fluxul  2       =    -0.0709

H_put_precedent =    21.1082
H_put_calc      =    21.3501
Delta h 1       =     0.0197
Fluxul  1       =    -0.0985


Iteratia nr.  5

Fluxul total pe putul imperfect 1 =    -1.4265
H_put_precedent =    21.8894
H_put_calc      =    21.8738
Delta h 6       =     0.2524
Fluxul  6       =    -0.1647

H_put_precedent =    21.6977
H_put_calc      =    21.6501
Delta h 5       =     0.1951
Fluxul  5       =    -0.1214

H_put_precedent =    21.5570
H_put_calc      =    21.4776
Delta h 4       =     0.1497
Fluxul  4       =    -0.1056

H_put_precedent =    21.4555
H_put_calc      =    21.3481
Delta h 3       =     0.1093
Fluxul  3       =    -0.0967

H_put_precedent =    21.3868
H_put_calc      =    21.2577
Delta h 2       =     0.0715
Fluxul  2       =    -0.0923

H_put_precedent =    21.3501
H_put_calc      =    21.2087
Delta h 1       =     0.0265
Fluxul  1       =    -0.1326
```

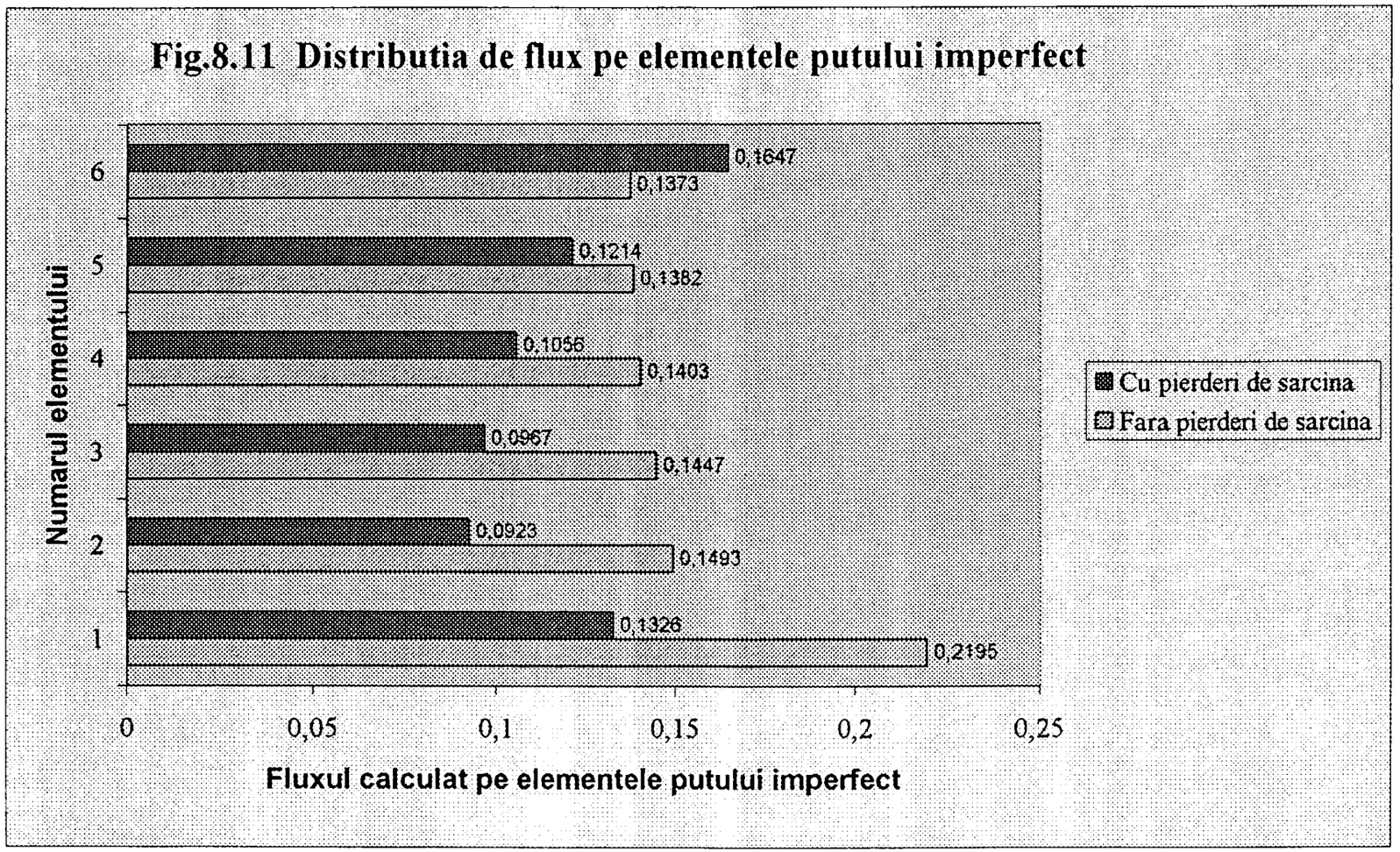


Fig.8.11 Distributia de flux pe elementele putului imperfect
Numarul elementului
6
5
4
3
2
1
0,1647
0,1373
0,1214
0,1382
0,1056
0,1403
0,0967
0,1447
0,0923
0,1493
0,1326
0,2195
0
0,05
0,1
0,15
0,2
0,25
Fluxul calculat pe elementele putului imperfect
Cu pierderi de sarcina
Fara pierderi de sarcina

CAPITOLUL IX

CONCLUZII

Una din resursele importante pentru alimentarea cu apă potabilă o constituie apa subterană. Față de alte surse de apă cele subterane se evidențiază mai ales prin calitatea deosebită pe care o au față de surdele de suprafață.

Concomitent cu intensificarea exploatării acestor surse de apă subterană s-au dezvoltat şi o serie de modele matematice şi numerice pentru optimizarea exploatării acestora.

Metodele pentru calculul exploatarea şi gospodărirea apelor subterane au evoluat foarte mult în ultimul timp prin introducerea şi folosirea pe scară largă a modelelor numerice de calcul. Aceste modele numerice au cunoscut o dezvoltare deosebită odată cu evoluția tehnicii de calcul.

În prezent, pe plan mondial există o serie de programe de calcul care în principal au la bază metoda diferențelor finite a volumelor finite şi a elementelor finite. În ultimul timp a început să se aplice şi metoda elementelor de frontieră şi cea a elementelor analitice însă fără să existe încă un program specific ca şi în cazul celorlalte metode.

Din concluziile şi comentariile efectuate în cadrul capitolului IV se poate observa că programele elaborate pe baza metodelor numerice standard (MEDIF, MEVFIN, MEFIN) prezintă dificultăți în ceea ce priveşte reprezentarea mişcării reale în vecinătatea elementelor de captare/infiltrare cum sunt puțurile, drenurile de lungime finită respectiv puțurile imperfecte. În aceste zone locale, metodele numerice standard pot conduce la erori semnificative şi care nu pot fi riguros controlate.

Analiza erorilor pe care le comportă metodele numerice în reprezentarea mişcării a fost efectuată prin folosirea unor variante de calcul cu diferite grade de discretizare a domeniului. Valorile obținute prin rularea modelelor numerice au fost comparate cu soluțiile analitice. Astfel s-au putut desprinde concluzii importante în ceea ce priveşte discretizarea domeniului şi erorile pe care aceste metode numerice le comportă.

Astfel, pentru metodele standard (MEDIF, MEVFIN, MEFIN) îndesirea rețelelor de discretizare reduce erorile care însă rămân semnificative şi relativ mari chiar şi pentru rețele dense de discretizare. În general mărirea gradului de discretizare conduce implicit şi la creşterea numărului de elemente ceea ce conduce în cele din urmă la mărirea volumului de date şi la mărirea dimensiunii sistemului de ecuații necesar rezolvării problemei în cauză. În general structura programelor pe

care rulează aceste metode standard are un volum limitat de date de intrare ceea ce impune o limită superioară a discretizării domeniului.

Superioritatea metodei elementelor de frontieră poate fi uşor observată prin precizia rezultatelor pe care aceasta le oferă față de metodele standard. După cum se poate observa erorile pe care această metodă le comportă se situează de regulă sub 1% sau în jurul a 1%. Metoda elementelor de frontieră îşi demonstrează superioritatea nu numai prin precizia rezultatelor ci şi prin tipul rezultatelor oferite. Astfel, folosind metoda elementelor de frontieră pentru drenuri, puțuri imperfecte se poate obține distribuția fluxului (distribuția de debit) pe elementele constituente ale acestor obiecte.

Pentru aplicarea metodei elementelor de frontieră în cadrul sistemelor complexe ale captărilor subterane s-a impus realizarea unui program de calcul complex (programul MEFRO) în cadrul căruia să poată fi introduse obiectele unui sistem complex de exploatare a apelor subterane. În acest sens pentru realizarea practică a programului s-a apelat la limbajul de programare Borland Pascal 7.0, limbaj de programare care dispune de mecanismele suport în ceea ce priveşte programarea structurată şi programarea orientării spre obiecte.

Folosirea programării orientate spre obiecte reprezintă un pas fundamental în elaborarea programului MEFRO. Prin folosirea adecvată a acestei tehnici de programare s-a putut realiza efectiv o implementare a tuturor obiectelor prezentate în capitolul V al tezei într-un singur program cadru foarte eficient.

O particularitate deosebit de importantă în elaborarea programului MEFRO a fost modul de concepere al meniurilor şi construcția de Unit-uri astfel încât fiecare Unit să îndeplinească o funcție precisă în cadrul programului general MEFRO. Realizarea meniurilor nu ar fi fost posibilă fără folosirea obiectelor Turbo Vision, care conțin funcții şi proceduri specifice realizării meniurilor programului MEFRO. Modul de folosire al obiectului TObject care reprezintă un obiect de bază în ierarhia Turbo Vision a fost esențial în realizarea meniurilor programului MEFRO şi mai ales în cadrul meniului de editare al datelor. Această importanță majoră derivă din faptul că toate obiectele standard Turbo Vision derivă din TObject mai puțin obiectele TPoint şi TRect.

Un alt scop pentru care s-a optat pentru programarea orientată spre obiecte a fost utilizarea obiectelor ca şi entități complete, nici un obiect neputând fi direct accesibil utilizatorului. Toate operațiile asupra câmpurilor sale efectuându-se doar prin intermediul metodelor. Metodele din cadrul obiectelor alcătuind astfel un set cât mai complet de operații relative la obiect.

Un alt aspect important în elaborarea programului MEFRO îl reprezintă folosirea structurilor dinamice de date. Fără folosirea unui limbaj de programare adecvat care să posede mecanismele de alocare dinamică a memoriei acest lucru nu ar fi putut fi posibil. Avantajele certe pe care structurile dinamice de date le oferă constau în faptul că oferă programatorului realizarea de programe în care zona de memorie a calculatorului să fie ocupată numai de anumite date necesare etapei de calcul în care se află programul la un moment dat. Astfel prin ingeniozitatea

programatorului se pot crea programe în care zona de memorie (Heap) să fie vehiculată de un flux de date optim. Acest lucru l-am realizat practic în carul programului MEFRO cu ajutorul obiectelor, constructorilor şi destructorilor.

Programarea structurată a permis crearea unor entităţi funcţionale bine conturate şi ierarhizate conform problemei. În cadrul acestor unităţi structurarea se manifestă atât la nivelul instrucţiunilor cât şi la nivelul datelor.

Modul de rezolvare a sistemului de ecuaţii a reprezentat de asemenea o preocupare majoră, aceasta pentru a evita o limitare strictă a volumului datelor de intrare. Soluţia finală adoptată constă în asamblarea matricei sistemului de ecuaţii şi stocarea ei pe hardisk-ul sistemului de calcul ceea ce practic crează posibilităţi enorme în ceea ce priveşte numărul ecuaţiilor pe care sistemul poate să le conţină. Această soluţie elimină complicaţiile legate de ocuparea memoriei calculatorului cu elementele matricei compacte a sistemului de ecuaţii.

Aplicaţiile programului MEFRO în cadrul sistemelor complexe ale captărilor subterane prezentate în cadrul capitolului VIII sunt rulările unor exemple standard pentru care a existat posibilitatea comparării rezultatelor rulării cu rezultatele exacte obţinute analitic.

Exemplele prezentate confirmă pe deplin aplicabilitatea programului, domeniul de aplicabilitate fiind mult mai larg decât cel prezentat în cadrul testărilor.

CUPRINS

BIBLIOGRAFIE

1	Abbott, M.B.	Computational Hydraulics, Pitman
2	Albu M.	Mecanica apelor subterane, Editura Tehnică, 1981
3	Banerjee, P.K. Butterfield, R.	Boundary Element Method in the Engineering Science. Mc. GRAW-Hill Book Company, London,New York (1981)
4	Bartha, I.	Curs de Hidraulică, Rotaprint, U.T. Iaşi, 1993
5	Bartha, I. Javgureanu V.	Hidraulică, Ed. Tehnică, Chişinău, 1989
6	Bateş, C.J.	A Computational Technique for the Efficient Handing of Large Matrices, „Int. J. Num. Methods eng.", No. 7, 1973
7	Bathe, K.J.	Finite Element Procedures in Engeneering Analysis, Prentice Hall, New Jersy, 1982.
8	Bathe, K.J. Wilson, F.L.	Numerical Methods in Finite Element Analysis, Prentice Hall, New Jersy, 1976.
9	Brebbia, C.A. Dominiquez, I.	Boundary Elements, Introductory Course Mc. GRAW-Hill Book Company. New York (1989)
10	Brebbia, C.A Telles, J.C.F. Wrobel, L.C.	Boundary Element Techniques, Springer, Berlin, New York, 1984
11	Charbeneau, R. Street, R. L.	Modelling Groundwater Flow Fields Containing Point Singularities – Water Resources Research, June 1979
12	Chung, T.J.	Finite Element Analysis in fluid Dynamics. New York: Mc GRAW-Hill, Inc, 1978
13	Cioc, D.	Hidraulică, Ed. Did. şi Ped. Bucureşti, 1983
14	Collaz, L.	The Numerical Treatment of Differential Equations, Springer Verlag, Berlin, 1966
15	Connor J.J. Brebbia C.A.	Finite Element Techniques for fluid folw – Butter worths London-Boston.
16	Corici, C. Mânz, D. Simulescu, A. Şerban M.	Limbajul PASCAL, Editura Libris, Cluj 1991
17	Creţu, V.	Structuri de date, I. P. Timişoara, 1987
18	Cristea, V. Kalisz, E. Athanasiu, I. Negreanu, L.	Borland Pascal 7.0, pentru Windows, Editura Teora, Bucureşti 1995.
19	Cristea, V. Kalisz, E. Athanasiu, L. Pănoiu, A.	Turbo Pascal 6.0, Editura Teora, Bucureşti 1992.
20	Dankert, J.	Numerische Methoden der Mechanic, VEB Fachbuch-Verlag, Leipzig, 1977

21	David, I.	Grundwasserfassungsanlagen anlagen mit Filterrohren. Technische Berichte. Institut fur Hidraulik und Hidrologie der TH Darmstadt, 1977, nr. 19
22	David, I.	La methode des elementes de frontiere indirecte dans la solution numerique de problemes d'infiltration dans les mileux poreux, Buletinul Ştiinţific şi Tehnic al Institutului Politehnic "Traian Vuia", Timişoara, 1988, Tom 33(47).
23	David, I.	Hidraulica, vol I+II, Universitatea „Politehnica" Timişoara, 1991
24	David, I.	Consideraţii asupra reprezentării puţurilor şi drenurilor de lungime finită în modelele numerice cu elemente finite. Simpozionul internaţional "Protecţia mediului, Amelioraţiile funciare şi folosirea energiei neconvenţionale în agricultură"21-22 mai 1992, Timişoara
25	David, I.	Grundwasserhydraulik, Studium Technik, Viweg, Germany, 1998
26	David, I. Eleş, G.	Realizarea unui pachet de programe "MEFIN"cu o structura interactiva pentru microsisteme PC/AT in domeniul hidraulicii subterane. Simpozionul international "Protectia mediului – Amelioratiile funciare si folosirea energiei neconventionale in agricultura -. 21-22 mai 1992
27	David, I. Eleş, G.	Studii cu privire la erorile introduse de către metoda elementelor finite în cazul singularităţilor. Sesiunea de comunicări ştiinţifice dezvoltare durabilă şi problemele mediului în spaţiul funciar, 15-16 mai, 1996, Bucureşti
28	David, I. Eleş, G.	Hydraulic Computational Approach of Partially Penetrated Wells, Buletinul Ştiinţific şi Tehnic al Universităţii „POLITEHNICA" din Timişoara, Seria Hidrotehnica, Tom 43(57), Fasc 2, 1998
29	David, I. P. Gaude	Relative error estimation for simulations of groundwater flow in the vicinity of partially penetrating wells using the Finite Volume Method, Buletinul Ştiinţific al Universităţii „POLITEHNICA" din Timişoara, Seria Hidrotehnica, Tom 43(57), 2000
30	David, I. Gerdes, H.	Incorporation of local three-dimensional flow in the plane BEM to model complex groundwater supply-system. Proceedings of the Boundary Element XVII. Computational Mechanics Publications Southampton – Boston, 1995
31	David, I. Gerdes, H.	Coupling Analytical, Boundary and Finite Element Methods to develop a model for groundwater flow with singularities generated by groundwater supply and recharge systems, Computational Mechanics in Water Resources XII – vol 1, Southampton, UK and Boston USA, 1998.
32	David, I. Şumălan, I.	Metode numerice în hidrotehnică, Ed. Mirton, 1998
33	David, I. Wehry, A. Man T.E.	Probleme actuale în tehnica drenajului, Editura Facla, Timişoara 1982.
34	Drobot. R	Bazele gospodăririi apelor I.C. Bucureşti, 1983
35	Drobot. R	Evaluarea resurselor de apă subterană din traseele râului Moldova în zona de debuşare a pârâului Ozana. Rev. Hidrotehnica, Bucureşti, vol 31, nr.4, 1986
36	Dodescu, Gh. Toma, M.	Metode de calcul numeric, Ed. Didactică şi Pedagogică, Bucureşti 1976.

37	Dorn, W.S.	Metode numerice cu programare în FORTRAN, Editura Tehnică, Bucureşti 1982.
38	Hâncu S. şi colaboratorii	Hidraulică aplicată, (partea III), Ed. Tehnică, 1985
39	Huisman, L. Olsthoorn, T-H	Artificial Groundwater Recharge. Pitman Advanced Publishing Program. Boston, London, Melborne, 1982
40	Kinzelbach, W.	Groundwater Modelling. An introduction with Sample Programs in BASIC. Amsterdam-Oxford-NewYork-Tokio: Elsevier 1986
41	Kinzelbach, W.	Aquifer Simullation Modell, A.S.M., Kasel Stuttgart, 1993
42	Kinzelbach, W. Rauch, R.	Aquifer Simullation Modell „ASM" – Dokumentation, Gesamthochschule Kassel-Universität, Kassel 1989
43	Klenke, M.	Numerische Modelltechink in der Grundwasserhidrologie. Institut für Wasserwirtschaft, Hidrologie und Landwirtschaftlichen Wasserbau, uni Hannover, Mitteilungen, Heft 59, 1986
44	Lanczos, C.	Solution of systems of linear equation by minimized iterations, „J. Pres. Nat. Bur Standards", 49 1952.
45	Ligett, I. Liu, Ph.	The Boundary Integral Equation Method for Porous Media Flow. George Allen & UNWIN, London 1983.
46	Marinescu, Gh. Rizzoli, I. Popescu, I. Ştefan, C.	Probleme de analiză numerică rezolvate cu calculatorul, Ed. Academiei R.S.R., Bucureşti, 1984.
47	Mihu, C.	Metode numerice în algebra liniară, Ed. Tehnică, Bucureşti, 1977
48	Moraru, F. Atodiroaiei, M.	Programarea microcalculatoarelor în sisteme de operare, Editura Ştiinţifică şi Enciclopedică, 1989
49	Muskat M.	The Flow of Homogenous Fluids through Porous Media, 1937
50	Pascariu, I.	Elemente finite. Concepte şi aplicaţii, Ed. Militară, Bucureşti, 1985
51	Pietraru, V.	Calculul infiltraţiilor, Ed. Ceres, Bucureşti, 1973
52	Pietraru, V.	Calculul infiltraţiilor, Ed. Ceres, Bucureşti, 1977
53	Pinder, F.G. Gray, W.	Finite Element Simulation in Surface and Subsurface Hydrology, Academic Press, New York, London, 1977
54	Schmith G.	Simulation von Grundwasserstromungen und Stofftransport-Matematische Modellierung Ruhr-Universitat, Bocum, 1987
55	Segui W.J.	Computer programs for the solution of System of linear Algebric-Equations, „Int. J. Num. Methods Eng.", No 7, 1973
56	Stematiu, D.	Calculul structurilor hidrotehnice prin metoda elementelor finite, Editura Tehnică, 1988
57	Vatnaskil Consulting Engineering	AQUA- User's guide, 1989, Iceland
58	Vraciu, G. Popa, A.	Metode numerice cu aplicaţii în tehnica de calcul, Editura Scrisul românesc, Craiova 1982.
59	Waterloo Hidrologic Inc.	Visual MODFLOW User's Manual, 1999
60	Wirth, N.	Algorithms and data structures, New Jersy, USA, 1986
61	Zienkiewicz, O.C Taylor, R.L.	The Finite Element Method Vol I and II Mc. GRAW- Hill Book Company, New York, 1989